THE OPTIMIZATION OF ENERGY RECOVERY SYSTEMS

By

JIGAR V. SHAH

A DISSERTATION PRESENTED TO THE GRADUATE COUNCIL OF
THE UNIVERSITY OF FLORIDA
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

1978

Dedicated to

my parents, Hemlata and Vinubhai Shah

ACKNOWLEDGEMENTS

The author wishes to express his sincere gratitude to Dr. A. W. Westerberg. Working as a graduate student under his supervision was an invaluable experience.

The author also wishes to express his appreciation to the department of chemical engineering at the University of Florida and to Dr. J. P. O'Connell and Dr. H. D. Ratliff of the supervisory committee.

Thanks are due to the assistance provided by the department of chemical engineering at Carnegie-Mellon University and to the Computer-Aided Design Centre, Cambridge, U.K.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF SYMBOLS

$\underline{a}$ = vector; constant for points on a hyperplane

b = scalar; constant for points on a hyperplane

C = the set of all newly violated constraints

C_1 = the violated constraint encountered first

C_p = heat capacity

D = minimum allowable approach temperature

E_0 = the equation set without any inequality constraints present as equalities (heat and material balances only)

$\hat{f}$ = scalar return function; may be subscripted

f_j = subproblem, subsystem, return function

F = flow rate

$\hat{\underline{G}}$ = matrix; first n rows are $(\underline{g}^1,\ldots\underline{g}^{n+1})$ and $(n+1)^{st}$ row is $(1,\ldots 1)$

$\underline{g}$ = vector constrained function; may be subscripted to represent scalar elements. May also be superscripted as $\underline{g}^i$ to represent the vector constraint function at a point i

h = enthalpy

H = a set of (m+1) constraints where m is an integer

$\hat{H}$ = a hyperplane

$I_{(j)}$ = the set of i such that stream i is an input to subsystem j

L = Lagrange function

LB = lower bound

$O_{(j)}$ = the set of i such that stream i is an output of subsystem j

Q = duty of a heat-exchanger

$\underline{q}$ = composite vector variable ($\underline{x}/\underline{u}$): may be subscripted

S = constrained variable set; may be subscripted

T = temperature

$\Delta T_{\lambda m}$ = log mean temperature difference

$\underline{u}$ = system decision variable; may be subscripted

U = heat transfer coefficient

UB = upper bound

V_c = the current set of inequality constraints in the equation set ($E_c - E_0$)

$V_1, V_2 \ldots V_{m+1}$ = subsets formed by removing one constraint at a time from H. (For example, V_1 is obtained by removing the first constraint in H.)

V_R = the set of constraints present in the equation set as equality constraints with the difference that their slack variables are used as search coordinates

V_S = all the constraints in the system less the ones in V_T and V_R

V_T = the set of constraints being held as equality constraints

$\underline{x}$ = vector variable associated with interconnecting streams; may be subscripted

$\underline{y}$ = multipliers to express a vector as a linear combination of other vectors; may be subscripted

Z = objective function for the linear programming problem (maximization)

Greek Letters

$\underline{\alpha}$ = variable vector used in linear programming formulation; may be superscripted to represent scalar variables in linear programming formulation

The scalar variable α and the subscripted variable α_{ij} may be used to represent a structural parameter (split fraction).

$\emptyset$ = objective function; may be superscripted as $\emptyset^i$ to represent its value at point i. May also be used as $\underline{\emptyset}$ to denote a vector of $\emptyset^i$'s

$\underline{\lambda}$ = vector of Lagrange variables; may be subscripted to represent scalar elements. May also be superscripted as $\underline{\lambda}^i$ to represent its value at point i

σ = slack variable; may be subscripted

Mathematical Symbols

$\cap$ = intersection with

'$\emptyset$' = null set

Abstract of Dissertation Presented to the Graduate Council
of the University of Florida in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy

THE OPTIMIZATION OF ENERGY RECOVERY SYSTEMS

by

Jigar V. Shah

March, 1978

Chairman: Arthur W. Westerberg
Major Department: Chemical Engineering

An emphasis on technical development oriented towards conservation of energy in the design of new chemical engineering processes and in the modification of the existing processes is as important as an emphasis on the search for alternate sources of energy. With the increasing sophistication in computer hardware, the design engineers are now able to assess innumerable options through the use of flowsheeting packages to maximize energy recovery. Most of the current flowsheeting packages essentially convert the well-defined input information about a process into a description about the output from the process. A next evolution in these packages is judged to be a system which will be very flexible in the input that is specified to it. In addition, it will possess an optimization capacity to yield the most desirable values for parameters left to it as degrees of freedom. This study describes a prototype for what a really convenient flowsheeting system ought to be.

The program EROS (Energy Recovery Optimization System) is a flow-sheeting package for evaluating and optimizing the performance of simple networks of heat-exchangers. Using EROS one can set up an arbitrary structure of heat-exchangers, stream splitters and mixers. Stream flow-rates and entry and exit temperatures may be specified, free, or bounded above and/or below. Phase changes may be allowed to occur. Requiring no more user input (such as initial guesses), the program gathers together the modeling equations and appropriate inequality constraints. It then develops solution procedures repeatedly in the course of optimizing, initially to aid in locating a feasible point (which need not be specified by the user) and in the final stages to take advantage of tight inequality constraints to reduce the degree of freedom. The program is written in standard Fortran and is currently operable on IBM 360/67.

The development of this program is relevant from the point of view of evaluating and optimizing energy recovery systems because the subject of choosing an optimal structure has recently come under increasing attention. A tool such as EROS will prove very useful to make a more thorough analysis of the candidate structures chosen by the process of synthesis at the penultimate and final stages. Existing energy recovery schemes may be reevaluated with the intention of making changes to make them more efficient. EROS can also perform quick reliability studies accounting for fluctuations in stream flow-rates and temperatures, a common problem during start-up and periodic failures.

As is the case in the optimization of most chemical engineering systems, a global optimum cannot always be guaranteed. An attempt

to recognize whether the discovered optimum is a global one, in a large system, is not a trivial task. Usually the problem may be side-stepped by making several optimization runs from different starting points and considering the best result as the required optimum. A systematic procedure is presented in this study to investigate global optimality, and the results on application to simple examples prove it to be quick and effective.

The use of EROS can be extended to perform process synthesis via structural parameters. However, the presence of inequality constraints inherent in the system can easily force the optimization to stop short of the desired result. This observation has not been reported before and attention is brought to it in this dissertation.

CHAPTER 1
INTRODUCTION

Most of the existing flowsheeting packages for chemical engineering processes are based on Sequential or Simultaneous Modular Approaches. In these approaches each unit is modeled by writing a computer subroutine which converts the input stream and equipment parameter values into output stream values. Systems based on Sequential or Simultaneous Modular Approaches are relatively easy to build but the penalties paid are the lack of flexibility in the definition of the problem and the requirement for well-defined user specifications.

Equation solving approaches, as followed in EROS (and in Leigh, Jackson and Sargent (1974), Hutchison and Shewchuk (1974) and Kubicek, Hlavacek and Prochaska (1976)) present an alternative way to treat flowsheets. The flowsheet is represented as a collection of non-linear equations which must be solved simultaneously. In following an Equation Solving Approach, the user can specify many of the values for both unit inputs and outputs. The unit equipment parameters are then calculated to give these desired transformations of inputs to outputs by the unit; in other words, the unit is designed to meet these requirements. Since EROS also contains an optimization capability the rather striking advantage is that variables for whose values the user has no preference for are treated as the degree of freedom by the system. EROS incorporates the general optimization strategy as outlined in Westerberg

and deBrosse (1973) and demonstrates the applicability and effectiveness of their algorithm.

The two important problems in the design of energy recovery systems are to choose the configuration, and, given a configuration, to choose the design parameters and operating variables. A recent review on the effort directed to choosing a configuration can be found in Nishida, Liu and Lapidus (1977). In choosing a suitable configuration the general trend has been to evaluate networks using the heuristic of setting the minimum allowable approach temperature to 20°F, whereas the economics often advocate a smaller value. In addition, when a stream is split, the need for finding the optimal value for the split fraction has been ignored. Grossman and Sargent (1977) optimized several heat-exchanger networks and found considerable savings (sometimes as much as 25%).

The problem of optimizing a heat-exchanger network to obtain the most suitable values for operating variables has been considered in the works of Westbrook (1961), Boas (1963), Fan and Wang (1964), Bragin (1966) and Avriel and Williams (1971). Typically each design problem is formulated and solved for as an optimization problem. Many investigators (Hwa (1965), Takamatsu, Hashimoto and Ohno (1970), Henley and Williams (1973), and Takamatsu et al. (1976)) have combined both the problems, choosing a configuration as well as the operating variables, and formulated it as an optimization problem.

All the methods mentioned for optimizing over the operating variables require the problem to be cast into a mathematical format. EROS precludes this need because of its capability as a flowsheeting package.

In Chapter II of the dissertation a description of the program EROS is provided. The data specification and modeling considerations for the system are discussed here while a user's guide to EROS is presented in the appendices. The derivation of a solution procedure is illustrated with the help of a simple example. If the user has not provided a feasible starting point, EROS has the capability of discovering one. The algorithm used is a modified version of that presented in deBrosse and Westerberg (1973). The optimization strategy based on Westerberg and deBrosse also merits attention in this study because of its usefulness to this application. The different applications of EROS and the results on optimization of 10 problems of varying sizes are also shown.

The program EROS does not guarantee a global optimum because of the non-linear and multimodal nature of the problem. In Chapter III an algorithm is presented to establish, often quite quickly, whether the discovered optimum is global or not. The strategy is based on finding both a lower (dual) bound for the optimum of the structured system and an upper bound on this lower bound. In this chapter the effectiveness of the algorithm is then illustrated by applying it to three small heat-exchanger network problems.

With a convenient evaluation and optimization package such as EROS, it is very tempting to use it to perform process synthesis via the use of structural parameters. Using this approach several alternate configurations are imbedded into one main structure which, on optimization, yields the required test configuration.

Several authors, as mentioned earlier in this section, have already experimented with this idea. However, the problem to be solved has to be formulated very carefully, and not enough attention has been paid to this fact. In Chapter IV a discussion regarding this subject is presented.

The conclusions from creating a system such as EROS and recommendations for further investigations form Chapter V of this dissertation. A guide to the use of the program is given in the appendices. The aspects of EROS presented are the input data format, the interpretation of the output, a typical computer printout, and a description of all the subroutines.

CHAPTER II
THE SYSTEM EROS

II.1. Background

In order to evaluate and optimize heat-exchanger networks it is desirable to have a program which on being given information about the configuration and stream properties, yields all the required information about the optimal network. Since the program will perform several different tasks, it would be very attractive and in many instances necessary for it to possess the following features.

a) An ability to set up solution procedures.
The program should eliminate or reduce computational recycles, choose the decision variables and discover the order for calculating the various unknown variables.

b) An ability to obtain a feasible starting point.
Often, locating a feasible starting point for optimization is not a simple task. In order to save users the time and trouble necessary to find a feasible starting point, the program should be capable of performing such a task on its own.

c) Efficient optimization routines.
These would be required for selecting the optimal values for the decision variables by searching over their full range.

However, the optimization of a system such as this one raises certain problems. The objective function is highly non-linear and multimodal. Also, if phase changes are allowed, continuous derivatives cannot be obtained. These criteria force the use of a search algorithm such as the complex method. In having resigned to the use of the

complex method for optimization, further demands can be made to improve the efficiency of the approach for optimization. If in the process of optimization several constraints are violated, one remedial action is the use of penalty functions, but this modification is inefficient and it increases the number of iterations required for convergence.

Hence, with regard to optimization, few additional features would be deemed attractive.

d) The ability to rederive a solution procedure.
When a constraint is violated, the program should be able to modify the equation set and rederive a solution procedure so that the optimization may be continued.

This strategy will lessen the number of iterations required for convergence as compared with the penalty function method. However, it is essential that the saving in computer time thus incurred compensates for the extra time required in rederiving the solution procedure.

e) The use of restriction as a solution strategy.
In the presence of a large number of decision variables, some of them are set to zero and optimization is performed with only the remaining ones as search coordinates. Optimization is considered complete when the Kuhn-Tucker (1951) optimality conditions are satisfied with respect to the decision variables held at zero value. This strategy aids considerably in reducing the number of iterations during optimization.

EROS is the author's attempt at developing an optimization program which incorporates all the features discussed above. Figure 1 illustrates the general structure of the EROS system.

The approach taken is to model each unit in a heat-exchanger network functionally by writing overall material and energy balances. The unit models themselves may be very complex internally, but the

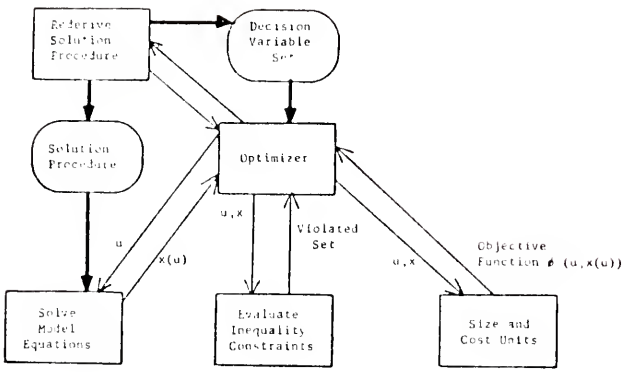


Fig. 1. Structure of an Optimizing System

net effect at the flowsheet level is that each unit satisfies overall heat and material balances. In the current version of EROS only simple models are used. Using the functional equations, the modeling of such a system is only at the flowsheet level, and a solution procedure or the order in which these equations are to be solved is found along with the degree of freedom to be chosen. The solution procedure sought is one that will eliminate, if possible, computational recycles at this level.

The above approach is useful because the equations are solved repeatedly as an inner loop to an optimization program. As illustrated in Figure 1, the optimizer directs all the activity. Its primary function is to adjust the decision variables to improve the objective function $\emptyset$. For this system $\emptyset$ is the annualized cost of the equipment plus the annual cost of buying the utilities needed such as steam for the purpose of heating.

To evaluate $\emptyset$, the optimizer supplies the block labeled "Solve Model Equations" with the values it wishes to try for the decision variables u. The remaining problem variables x(u) are then obtained by solving the model equations. With u and x(u) values available, constraint violations are checked, and if some are violated, they are identified to the optimizer. Assuming none are violated, the units in the system are sized and an annualized cost $\emptyset$ is evaluated. The optimizer notes this cost and changes the decision variable values, with the aim of reducing $\emptyset$. This calculation sequence is repeated many times during a typical optimization. If constraints are violated, special action is taken, which for this system will result in a modified set of model equations and a need to rederive a solution procedure

for them (this approach is based on the optimization strategies in deBrosse and Westerberg (1973) and Westerberg and deBrosse (1973)). The modified complex optimization algorithm (Umeda and Ichikawa (1971)) is used in searching for the decision variable values, u.

II.2. Data Specification

Adequate data must be supplied to the computer to define the problem. A problem definition will require the following input.

1) the flowsheet structure
   a) connectivity of segments and units

2) the unit data
   a) unit type
   b) any equipment parameter specifications

3) the stream data
   a) flow and temperature specifications
   b) physical property data
   c) film heat transfer coefficients
   d) materials specifications and other cost parameters

4) the segment data
   a) associated stream identifier (i.e., what stream this segment is a part of)
   b) any specifications imposed on flow and temperature

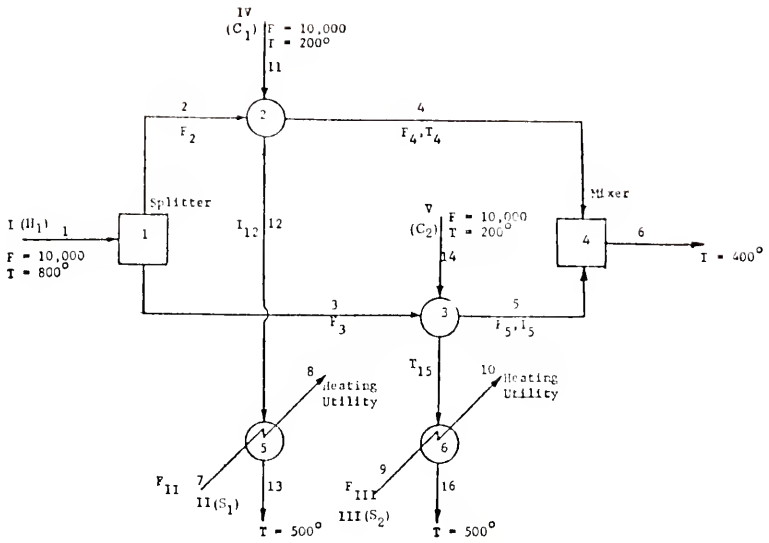
5) general user specifications

6) guessed set of inequality constraints to be held

II.3. Modeling Considerations

The modeling of a network will be illustrated with regard to the example in Figure 2.

The network comprises a single hot process stream H_1 which is split and used to heat two cold process streams C_1 and C_2. It then merges to its exit conditions. Streams C_1 and C_2 are heated further by steam utility streams S_1 and S_2. The network has four heat



$C_p = 1$ $U_2 = 25$ $U_3 = 20$ $U_5 = U_6 = 50$

For Stream II $T_{in} = T_{out} = 600°$ Heat of Vaporization = 800

Cost multiplier per unit flow rate = 0.4

For Stream III $T_{in} = T_{out} = 600°$ Heat of Vaporization = 800

Cost multiplier per unit flow rate = 0.2

Constraints: $2500 \leq F_2, F_3 \leq 7500$, $300° \leq T_{12}, T_{15} \leq 500°$

Aim: To minimize ϕ. $\phi = \sum_{i=2,\ i\neq 4}^{6} \text{Cost}_i + 0.4\, F_{II} + 0.2\, F_{III}$

where $\text{Cost}_i = 35\ (\text{Area}_i)^{0.6}$. Cost_i and Area_i are associated with exchanger i.

Fig. 2. An Example Problem

exchanger (or heater and cooler) units, 2, 3, 5 and 6, one stream splitter unit, 1, and one mixing unit, 4. These units may also be referred to as nodes. All the heat exchangers are assumed counter-current. The streams have been broken up into segments, of which there are 16 overall. For example, stream C_1 enters node 2 as segment 11. It exits and proceeds to unit 5 as segment 12, and finally leaves the system as segment 13. The naming scheme should now be evident. The 3 basic units used are shown in Figure 3. The unit models are written functionally by writing overall material and energy balances.

Unit

1

$$F_2 = \alpha F_1 \tag{1}$$

$$F_3 = (1 - \alpha)F_1 \tag{2}$$

$$h_2 = h_1 \tag{3}$$

$$h_3 = h_1 \tag{4}$$

2

$$F_4 = F_2 \tag{5}$$

$$F_{12} = F_{11} \tag{6}$$

$$h_2F_2 + h_{11}F_{11} = h_4F_4 + h_{12}F_{12} \tag{7}$$

3

$$F_5 = F_3 \tag{8}$$

$$F_{15} = F_{14} \tag{9}$$

$$h_3F_3 + h_{14}F_{14} = h_5F_5 + h_{15}F_{15} \tag{10}$$

4

$$F_6 = F_4 + F_5 \tag{11}$$

$$h_6F_6 = h_4F_4 + h_5F_5 \tag{12}$$

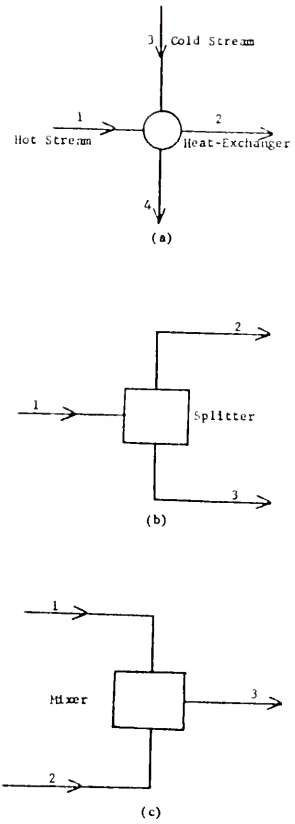


Fig. 3. The Network Nodes

5 $$F_8 = F_7 \tag{13}$$

$$F_{13} = F_{12} \tag{14}$$

$$h_7F_7 + h_{12}F_{12} = h_8F_8 + h_{13}F_{13} \tag{15}$$

6 $$F_{10} = F_9 \tag{16}$$

$$F_{16} = F_{15} \tag{17}$$

$$h_9F_9 + h_{15}F_{15} = h_{10}F_{10} + h_{16}F_{16} \tag{18}$$

In addition to these 18 equations, the associated inequality constraints and the equipment sizing and costing relations can be written.

The basic inequality constraint is that at no point in the exchanger should the hot stream temperature equal or fall below that of the cold stream. Referring to the heat-exchanger in Figure 3a, this constraint is usually expressed as

$T_1 \geq T_4 + D$, where D is minimum allowable approach temperature.

$T_2 \geq T_3 + D$

However the temperatures could cross over internally and these constraints may not be adequate to detect it, particularly when a stream passes through a phase change. A check should therefore be made at several points along the exchanger to prevent a "crossover" of temperatures.

The final set of constraints indicate that a positive heat transfer must occur, that is, the hot stream must be cooled and the cold stream heated. These are

$$T_2 \leq T_1 \quad \text{and} \quad T_3 \leq T_4 \ .$$

All the constraints associated with a typical exchanger such as the one shown in Figure 4 are listed in Table 1 with their appropriate code so that they can be precisely identified by numbers. No constraints are written for the splitter and mixer units.

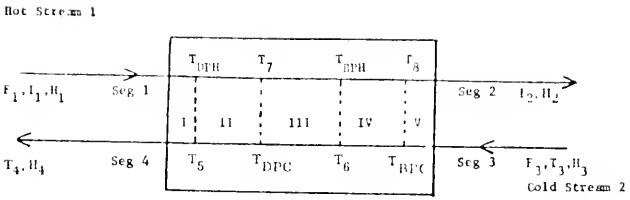
The sizing calculation for an exchanger is to evaluate its area. This calculation can be very involved, but for design purposes may in fact be simplified by assuming film coefficients based on the fluid, whether it is heating or cooling and whether it is boiling or condensing (see Perry and Chilton (1973)). The exchanger may, again, for simplified design purposes, be considered to operate in zones as indicated in Figure 4. Each zone is then sized and costed using the appropriate film coefficients and temperature driving forces.

To place a crude cost on the exchanger one might use an equation of the form (see Guthrie (1969)).

$$\text{cost} = f_M f_P (aA^m)$$

Where f_M is a materials factor, f_p a pressure factor and A the area of a zone within an exchanger. The zones are sized and costed differently because of the different types of heat-exchange duty. The terms a and m are constants, with m being about 0.6 to 0.8. Constant costs are assumed for the splitter and the mixer units.

The last source of equations is the evaluation of physical properties. The system must be able to convert from stream temperature (and vapor fraction for a pure component in two phase region) to enthalpy and vice versa. A cooling curve may be provided for the stream



T_{DFH} = Dew point temperature, hot stream

T_{BFH} = Bubble point temperature, hot stream

T_{DFC} = Dew point temperature, cold stream

T_{BFC} = Bubble point temperature, cold stream

T_5 = Temperature of the cold stream at a location where the hot stream temperature is T_{DFH}

T_6 = Temperature of the cold stream at a location where the hot stream temperature is T_{BFH}

T_7 = Temperature of the hot stream at a location where the cold stream temperature is T_{DFC}

T_8 = Temperature of the hot stream at a location where the cold stream temperature is T_{BFC}

D = Minimum allowable approach temperature

Fig. 4. Partitioning a Heat-exchanger into Zones where Phase Changes Occur

TABLE 1

CONSTRAINT REPRESENTATION FOR AN EXAMPLE NODE 3 IN FIGURE 4

Comment	Exterior Constraints	Code
approach	$T_1 \geq T_4 + D$	(NODE * 10) + 1 = 31
approach	$T_2 \geq T_3 + D$	(NODE * 10) + 2 = 32
	$T_1 \geq T_2$	(NODE * 10) + 3 = 33
	$T_4 \geq T_3$	(NODE * 10) + 4 = 34
F_{1LB} = lower bound for flow F_1	$F_1 \geq F_{LB}$	(NODE * 10) + 5 = 35
F_{1UB} = upper bound for flow F_1	$F_{UB} \geq F_1$	(NODE * 10) + 6 = 36
F_{3LB} = lower bound for flow F	$F_3 \geq F_{3LB}$	(NODE * 10) + 7 = 37
F_{3UB} = upper bound for flow F	$F_{3UB} \geq F_3$	(NODE * 10) + 8 = 38

Type	Interior Constraints	Representation
approach	$T_8 \geq T_{BPC} + D$	(NODE * 1000) + 1 = 3001
approach	$T_{BPH} \geq T_6 + D$	(NODE * 1000) + 2 = 3002
approach	$T_7 \geq T_{DPC} + D$	(NODE * 1000) + 3 = 3003
approach	$T_{DPH} \geq T_5 + D$	(NODE * 1000) + 4 = 3004

In addition there could be constraints associated with any stream segment.

Example: Segment 3, Node 3

Comment	Constraints	Code
H_{3LB} = lower bound for enthalpy H_3	$H_3 \geq H_{3LB}$	- (SEG * 1000 + NODE * 10 + 1) =-3031
H_{3UB} = upper bound for enthalpy H_3	$H_{3UB} \geq H_3$	- (SEG * 1000 + NODE * 10 + 2) =-3032

if the stream is assumed to be at a constant pressure. Figure 5 illustrates a cooling, curve, where T_D and T_B are the dew and bubble point temperatures respectively.

Properties such as thermal conductivities and densities should also be provided if the film coefficients are to be determined from correlations. If for design purposes typical values for film coefficients are to be used, these properties will not be needed.

II.4. Deriving the Solution Procedure and Solving Model Equations

Consideration will now be given to developing a solution procedure and then solving the example problem. First, the system must gather together the necessary equations, or at least establish their structure, so that a solution procedure may be prepared. The desired solution procedure should eliminate all recycle loops in the computations if possible or minimize their number.

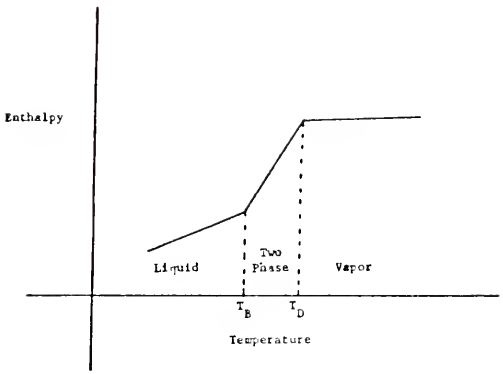
The initial solution procedure will ignore all but the user specified inequality constraints. Thus the system sets up the 18 heat and material relations shown in the last section. Assuming that the user has requested that constraints 55 and 33 be included, the additional relations required are

$$F_7 = F_{7LB} + \sigma_{55} \tag{19}$$

$$T_3 = T_5 + \sigma_{33} \tag{20}$$

where F_{7LB} is the lower bound on F_7.

The inequality constraints have been converted to equality by the slack variables σ_{55} and σ_{33} which are required to be nonnegative.



T_B = Bubble point temperature

T_D = Dew point temperature

Fig. 5. A Typical Cooling Curve

When the solution procedure is derived σ_{55} and σ_{33} will be required to be decision variables with an initial value of zero. In this way F_7 and T_3 will be forced to equal to F_{7LB} and T_5 respectively.

The relation (20) is in terms of temperatures rather than enthalpies. Hence the following relationships should also be added

$$T_3 = f(h_3) \tag{21}$$

$$T_5 = f(h_5) \tag{22}$$

The system can implicitly account for equations (20), (21) and (22) by the single expression

$$h_3 = f(h_5, \sigma_{33}) \quad . \tag{23}$$

The incidence matrix can now be created. However its size can be significantly reduced. Note that a large number of equations simply equate one variable to another. Equations (3), (4), (5), (6), (8), (9), (13), (14), (16) and (17) are precisely of this form. These equations will automatically be satisfied if the values of variables so equated are stored in a common storage location. Hence these equations may be deleted and the two variables occurring in each one of them may be merged to a single one.

Many of the variables in the incidence matrix are in fact specified. Let the following be specified in data input for the example in Figure 2

Flows $F_1, F_{11}, F_{14}, F_{7LB}$

Enthalpies $h_1, h_6, h_7, h_8, h_9, h_{10}, h_{11}, h_{13}, h_{14}, h_{16}$

These specified variables along with the slack variables σ_{55} and σ_{33} (required to be decision variables) may be eliminated in the incidence matrix. The resulting and much reduced matrix is illustrated in Table 2.

A modification of Lee, Christensen and Rudd (1966) algorithm is applied to determine the solution procedure. The solution procedure that results on the application of this algorithm to the incidence matrix of Table 2 is shown in Table 3. The variables listed are calculated from the corresponding equations in the order indicated. Note that there is an iteration loop involving the single 'tear' variable F_2 (from steps 4 to 9). $F_2(=F_4)$ appears in equation (12) and the iterations between steps 4 and 9 are continued until the value of F_2 guessed at step 4 is essentially the same as the value of F_2 calculated from equation (12) in step 9.

The execution of the solution procedure, that is, calculation of the variables from the equations assigned to them, is termed "Solve Model Equations" in Figure 1. Corresponding to every unit, a subroutine is required to calculate any variable involved in the heat and mass balances of the particular unit. These subroutines may be supplied by the user for more sophisticated models.

II.5. Starting the Problem: Finding a Feasible Point

If the user has not provided any information to aid in obtaining a feasible starting point, a modified version of an algorithm by deBrosse and Westerberg (1973) is used. As mentioned earlier, a significant effort would be required on the part of a user to provide a feasible starting point if computational loops are involved in solving

TABLE 2

THE INCIDENCE MATRIX

Equations \ Variables	$E_{7,8}$	$E_{9,10}$	$E_{2,4}$	$E_{3,5}$	E_{12}	E_4	E_5	E_{15}	E_6	α
(1)			X							X
(2)				X						X
(7)			X		X	X				
(10)				X			X	X		
(11)			X	X					X	
(12)			X	X		X	X		X	
(15)	X				X					
(18)		X						X		
(19)	X									
(23)							X			

TABLE 3

SOLUTION PROCEDURE FOR THE PROBLEM IN FIGURE 2

Decision Variables	σ_{55}, σ_{33}
Variable	Equation
1. $F_7 (=F_8)$	(19)
2. h_{12}	(15)
3. h_5	(23)
4. Guess $F_2 (=F_4)$	
5. α	(1)
6. $F_3 (=F_5)$	(2)
7. h_4	(7)
8. F_6	(11)
9. $F_2 (=F_4)$	(12)
10. H_{15}	(10)
11. $F_9 (=F_{10})$	(18)

model equations. Computational loops are almost inevitable in complex networks. However, the user does have the option of providing a feasible starting point.

The deBrosse and Westerberg (1973) algorithm uses an indirect approach. It hypothesizes that a subset of constraints has no feasible region and then attempts to verify the conjecture. If successful, the subset is identified as infeasible and obviously no feasible point exists. If unsuccessful, either a new hypothesis can be generated or the algorithm has indirectly found a feasible point.

The feasible point algorithm

In order to demonstrate the feasible point algorithm, the following definitions are necessary.

E_0 = The equation set without any inequality constraints present as equalities (heat and material balances only).

E_c = The current equation set (E_0 + inequality constraint(s) present as equalities with slack variables).

C = The set of all newly violated constraints.

C_1 = The violated constraint encountered first.

V_c = The current set of inequality constraints in the equation set: $(E_c - E_0)$.

H = A set of (m + 1) constraints.

$V_1, V_2, \ldots V_{m+1}$ = subsets formed by removing one constraint at a time from H (for example, V_1 is obtained by removing the first constraint in H).

'Ø' = null set.

Degree of freedom (as defined for this work) = The number of original variables (excluding slack variables) in problem less number of constraints in E_c.

Procedure 'A' represents the following sequence of operations.

1) Set all the decision variables to a value of zero.

2) Solve the model equations.

3) Check for constraint violations.

Step 1 Set $E_c = E_0$, $V_c = '\emptyset'$

Step 2 Determine the solution procedure based on structural considerations of the equation set. If a solution procedure can be determined, go to Step 4.

Step 3 Attempt unsuccessful. Exit.

Step 4 Execute Procedure 'A'. If any constraints are violated, go to Step 6.

Step 5 Feasible starting point. Exit.

Step 6 Set $E_c = [E_c, C_1]$

Step 7 Determine the solution procedure. If a solution procedure cannot be found, go to Step 11.

Step 8 Execute Procedure 'A'. If the problem does not prove solvable, go to Step 11. If no constraints are violated, go to Step 5.

Step 9 If the degree of freedom is not zero, go to Step 6.

Step 10 Set $H = [V_c, C_i]$, $i = 1$. Go to Step 12.

Step 11 Set $i = 1$, $H = V_c$

Step 12 Set $E_c = [E_0, V_i]$, $V_i \neq V_c$ from Step 10.

Step 13 Determine the solution procedure. If it cannot be determined, go to Step 17.

Step 14 Execute Procedure 'A'. If the problem cannot be solved go to Step 17. If no constraints are violated, go to Step 5.

Step 15 If $C \cap H \neq '\emptyset'$, go to Step 17.

Step 16 If the degree of freedom = 0, go to Step 10. If not, go to Step 6.

Step 17 Set $i = i + 1$. If $i > m + 1$, go to Step 3 (the set of constraints in H do not allow for a feasible region). If $i \leq m + 1$, go to Step 12.

The main departures from the deBrosse and Westerberg (1973) algorithm are as follows:

1) In this work, the set H is restricted to containing less than or equal to $(n + 1)$ constraints where n represents the number of decision variables.

2) Both the sets V_c and H are created in a different manner. In this work, one constraint is added at a time to build up the set V_c, and H is created either when there are already n constraints in V_c and an additional constraint is violated, or when a structural infeasibility occurs.

3) "No Solution" options (in Steps 8 and 14 where the problem is not solvable) are not treated in as rigorous a fashion as in deBrosse and Westerberg (1973). The method here is the same unless several problems corresponding to the same "hypothesis" set H lead to no solution in Step 14. The algorithm here may terminate unsuccessfully at Step 17 whereas that of deBrosse and Westerberg might continue with alternate and reduced sets H. This option is quite complex and was never found necessary in EROS. Hence it was never included.

The algorithm can now be applied to the example shown in Figure 2. (Note that this problem involves 8 equations in 10 unknowns; thus two degrees of freedom exist at the start.)

E_0 = [(1), (2), (7), (10), (11), (12), (15), (18)]

(all the equations from Table 2 except the last two, (19) and (23))

1) $E_c = E_0$, V_c = 'Ø'

2) Solution procedure derived

4) Procedure 'A' executed

Constraint Violation = 33.

6) $E_c = [E_0, 33]$

7) Solution procedure derived

8) Procedure 'A' executed

Constraint Violation = 25

9) Degree of freedom = 1

6) $E_c = [E_0, 33, 25]$

7) Solution procedure derived

8) Procedure 'A' executed

Constraint Violation = 22

9) Degree of freedom = 0

10) $H = [33, 25, 22]$, $V_1 = [33, 25]$, $V_2 = [22, 33]$, $V_3 = [22, 25]$

12) $E_c = [E_0, V_2] = [E_0, 22, 33]$

Note: $V_1 = [33, 25]$ is V_c in the previous iteration and it need not be considered again

13) Solution procedure derived

14) Procedure 'A' executed. Constraint Violation = 55

15) $C \cap H$ = '0'

16) Degree of freedom = 0

10) $H = [22, 33, 55]$, $V_1 = [22, 33]$, $V_2 = [55, 22]$, $V_3 = [55, 33]$

12) $E_c = [E_0, V_2] = [E_0, 55, 22]$

Note: $V_1 = [22, 33]$ is V_c in the previous iteration and is eliminated from consideration

13) Solution procedure derived

14) Procedure 'A' executed

No constraint violation

5) Feasible starting point. Exit.

II.6. The Optimization Strategy

The optimization strategy is modeled after the algorithm presented in Westerberg and deBrosse (1973). The algorithm is invoked once a feasible point is available.

The sets of inequality constraints are divided into three sets.

V_T = The set of inequality constraints being held as equality constraints.

V_R = The set of constraints present in the equation set as equality constraints with the difference that their slack variables are used as search coordinates.

V_S = The set of all remaining constraints.

V_c = $[V_T, V_R]$ (Note: V_c is as defined in the previous section.)

Solution procedures are modified as inequality constraints are moved from one set to another. Adding constraints in the set being held tends to aid the optimization process by reducing the dimension of search space for what is usually a marginal or no added burden in solving an enlarged set of equations.

As the optimization proceeds, the values of all variables, including all slack variables, are stored for the point that yields the best value for the objective function. Hence, even when V_c is changed, optimization can be and is started at the best point discovered up to that moment. This modification makes a significant improvement to the Westerberg and deBrosse (1973) method. Figure 6 illustrates the typical dilemma faced by the Westerberg and deBrosse optimization algorithm when stepping from a current best point, point 'e', through one or more inequality constraints to point 'f'. At point 'f' the

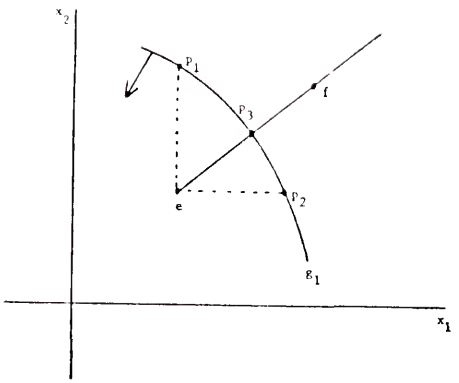


Fig. 6. Keeping Track of the Best Point 'e'

constraint g_1 is detected as being violated. The algorithm will respond by changing the solution procedure so that the slack variable σ_1 for g_1 becomes a decision variable. The other decision variable will be either x_1 or x_2. There are several options now as to where the optimization may be started. The algorithm could hold x_1 or x_2 (whichever is selected as the decision variable) at its current value and find the point where σ_1 is zero, leading to point P_1 or P_2 respectively. Alternatively it could attempt to locate P_3 by searching along the direction leading from 'e' to 'f'. All of these options can, and often do, lead to a next point which has a higher and thus worse value for the objective function. By saving all the variable values for the best point, the search can always start, even after developing a new solution procedure, from that point, that is from point 'e'. This change reduces cycling because a change in the solution procedure cannot lead to a point that is worse.

The actual search strategy used is the modified complex method (Umeda and Ichikawa (1971)). The complex method is considered suitable because gradients are not required. The treatment of phase changes creates discontinuities in first derivatives of functions.

The optimization Algorithm

The notation followed is the same as that in the last section. Figure 7 gives the structure of this algorithm.

Step 1 Obtain a feasible starting point.

Step 2 Set $V_T = V_c$, $V_R = \emptyset$'. Apply the Kuhn-Tucker (1951) conditions as follows. Perturb each slack variable

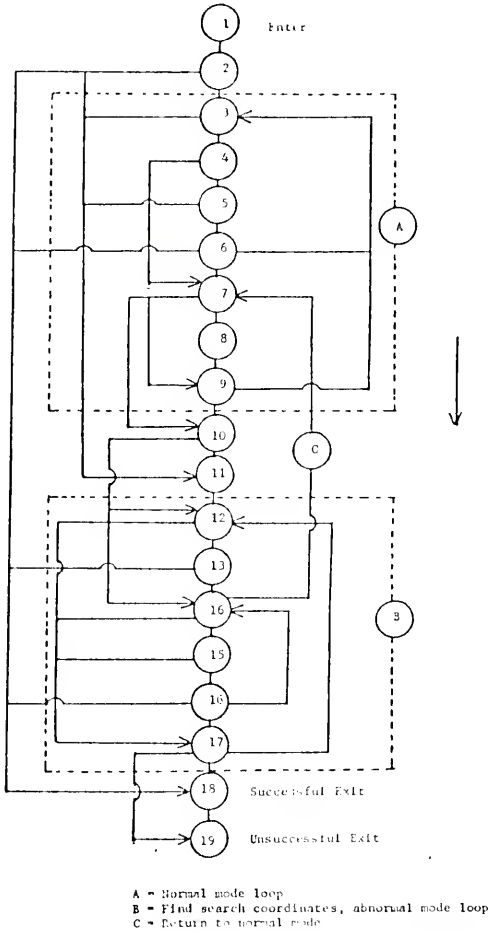


Fig. 7. Structure of the Optimization Algorithm

corresponding to constraints in V_T, one at a time, away from zero. If an improvement in the objective function results, move the corresponding constraint from V_T to V_R.

A. If any solution procedure fails numerically during this step, go to Step 11.

B. If on completion of this step, the number of constraints in V_T is equal to the number of constraints in V_c, and the degree of freedom is zero, go to Step 18.

C. Otherwise continue.

*The optimization is back in its normal mode if return is to Step 3 or Step 7.

Step 3 Perform an optimization (using the complex algorithm). Use as search variables the slack variables corresponding to inequality constraints in V_R along with any additional problem variables still retained as decision variables. Search only over nonnegative values of the slack variables.

A. If at any point the solution procedure fails numerically, go to Step 11.

B. Otherwise, when the optimization algorithm exits normally, continue.

Step 4 An optimization has just been completed.

A. If one or more constraints are violated, go to Step 7.

B. Otherwise continue.

Step 5 Set $V_{Told} = V_T$. Apply the Kuhn-Tucker conditions as done in Step 2.

A. If any solution procedure fails numerically, go to Step 11.

B. Otherwise on completion continue.

Step 6 A. If $V_T = V_{Told}$, go to Step 18.

B. Otherwise repeat from Step 3.

*One or more constraints not in the current set V_c has been violated.

Step 7 A. If the degree of freedom is not zero, set $E_c = [E_c, C_1]$ and go to Step 9.

B. Otherwise find the set of constraints R from V_R such that each constraint in R could, from structural considerations of the equations, be traded for C_1. If R = 'Ø', go to Step 10.

C. Find the constraint C', in set R having the largest value for its corresponding slack variable. If that value is zero, go to Step 10.

D. Otherwise continue.

Step 8 Replace C' with C_1 in V_R.

Step 9 For any slack variable corresponding to a constraint in V_R and having a value of zero at the current best point, transfer the corresponding constraint from V_R to V_T. Set $V_c = [V_R, V_T]$, $E_c = [E_0, V_c]$ and develop a new solution procedure.

A. If a solution procedure cannot be developed, go to Step 11.

B. Otherwise return to Step 3.

*The degree of freedom is zero and either (a) the values of all the slack variables in V_R are zero or (b) the set R is empty.

Step 10 Set $H = [V_C, C_1]$, set i = 1, and go to Step 12.

*The current set of equations are (or appear to be) numerically inconsistent.

Step 11 Set $H = [V_c]$, set i = 1.

Step 12 Set $E_c = [E_0, V_i]$, $V_i \neq V_c$ (this test is only relevant if entry was originally from Step 10 above). Set V_R equal to the set of all constraints in V_i whose slack variable values are greater than zero. Set V_T equal to all remaining constraints in V_i.

A. If V_R = 'Ø', go to Step 13.

B. Otherwise, determine a new solution procedure.

1. If one can be found, go to Step 14.

2. Otherwise, go to Step 17.

*If entry to Step 12 was via Step 10 originally, a degenerate problem is at hand as V_c and C_1 (N_{Vc} + 1 constraints) are all satisfied in a subspace of dimension N_{Vc}.

Step 13 Determine a solution procedure.

A. If one cannot be determined, go to Step 17.

B. Otherwise, perturb, one at a time, the slack variables corresponding to the constraints in V_T.

1. If, on any perturbation, constraints are violated such that $H \cap C \neq$ 'Ø', go to Step 17.

2. If, on any perturbation of a slack variable, an improvement is obtained in the objective function, put the corresponding constraint into V_R.

C. If, on completion of the perturbations in Step B, the set V_R = 'Ø', go to Step 18.

D. Otherwise continue.

Step 14 Perform an optimization. Use as search variables the slack variables corresponding to inequality constraints in V_R along with any additional problem variables still retained as decision variables. Use the solution procedure derived in Step 13. Search only over nonnegative values of the slack variables.

A. If at any point the solution procedure fails numerically, go to Step 17.

B. When the optimization algorithm exits normally and if a constraint violation occurs, then

1. If $H \cap C \neq '\emptyset'$, go to Step 17.

2. Otherwise, $H \cap C = '\emptyset'$. Go to Step 7.

C. Otherwise when the optimization algorithm exits normally and no constraints are violated, continue.

Step 15 Set $V_{Told} = V_T$. Apply the Kuhn-Tucker conditions (as done in Step 2).

A. If any solution procedure fails numerically, go to Step 17.

B. Otherwise continue.

Step 16A. If $V_T = V_{Told}$, go to Step 18.

B. Otherwise, return to Step 14.

*The current solution procedure failed or led to an immediate constraint violation of the constraint dropped in set H.

Step 17 Set $i = i + 1$.

A. If $i > m + 1$, go to Step 19.

B. Otherwise repeat from Step 12.

*Normal exit. Optimization attempt apparently successful.

Step 18 Optimization complete. Exit.

*Abnormal exit. Optimization attempt aborted.

Step 19 Optimization failed. Exit.

Application of the strategy to the Example Problem in Figure 2.

1) Feasible starting point. V_c = [55,22] (from the last section).

2) V_R = [22]

3 and 4) Optimization with the complex method.

7) Constraint violated = 35, R = [22], and $\sigma_{22} > 0$.

8) Replace constraint 22 with constraint 35.

9) V_R = [35], V_T = [55]. $E_c = [E_0, V_R, V_T]$.

3) Optimization with the complex method.

4) Optimization complete.

5 and 6) V_{Told} = [55]. V_T = [55] after Kuhn-Tucker test.

18) Optimization successful. Exit.

At the optimum, the following values resulted:

$F_2 = F_4 = 7350$, $F_3 = F_5 = 2650$, $T_{12} = 500°$, $T_{15} = 300°$,

$F_7 = 0$, $F_9 = 2500$, $T_4 = 392°$, $T_5 = 423°$ and $\emptyset = 3538$

II.7. Special Uses of EROS

Using existing Heat-Exchangers. The program may be used to analyze a network where some of the exchangers are already specified. A special effort must be made, however, to make use of these exchangers. It is assumed that these exchangers are available at no cost. In Figure 8, an exchanger with an area of A_1 has been specified. On analysis, however, it is discovered that an exchanger with an area A_2 is required at that particular site in the network. In the program, the costs assumed for different conditions of A_1 and A_2 are shown in Figure 8. The physical significance of 1) is that a by-pass will be

1) if $A_2 \leq A_1$ Cost = 0

2) if $A_2 > A_1$ Cost = Cost associated with an exchanger having the area $(A_2 - A_1)$

Fig. 8. Using an Existing Exchanger

used in the specified exchanger. 2) implies that an exchanger with area = $A_2 - A_1$ must be purchased in addition to the exchanger already present.

Reliability analysis. The program has been extended to permit its use in quick reliability studies. The reliability studies will be demonstrated with the help of an example. Figure 9a represents a network that is operational under normal conditions. Cold process streams C_1, C_2, and C_3 are heated to their final temperatures with the help of a hot process stream H_2 and a flue gas H_1. The flow rate and the outlet temperature of stream H_1 are undefined but are required to be within specified bounds. Assume that two abnormal occurrences take place separately, for certain periods of the year, namely,

1) Stream H_2 is unavailable.

2) Heating of stream C_2 is no longer required.

The aim now is to find an optimal network such as the one shown in Figure 9a, fully provided for to meet the contingencies with the aid of by-passes in the exchangers or with the aid of auxiliary exchangers. The designer is permitted to allow a change in the flow of stream H_1 and its outlet temperature provided they stay within their specified bounds. In the case of failure mode 2), a cooling utility stream may be used to cool stream C_2 to 230° so that it may be recycled to be heated to 250° if it is desired to maintain a semblance to the normal operation. (Flow rate of C_2 can be allowed to vary between 0 and 70,000 in this instance.)

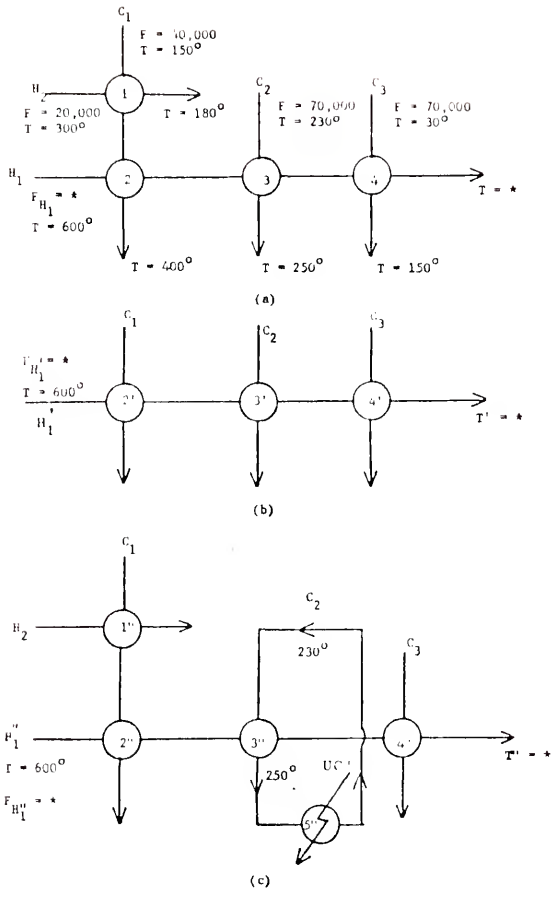


Fig. 9. Reliability Analysis

Figure 9b represents the network when stream H_2 is unavailable, and, Figure 9c when stream C_2 is not required to be heated. Additional user specifications to those shown in Figure 9 are presented in Table 4. In order to find the optimal operating system, networks in Figure 9a, 9b and 9c are optimized together. The objective function $\emptyset$ is given by

$$\emptyset = C_{H_1} F_{H_1} + C_{H_1'} F_{H_1'} + C_{H_1''} F_{H_1''} + C_{UC''} F_{UC''} + \sum_{i=1}^{5} (\text{cost of exchanger area at site i})$$

where C_i and F_i represent the cost coefficient and the flow rates of stream i, respectively. The cost coefficient C_i should reflect the expected fraction of the year that the network is in the particular state being represented. For example, for the problem in Figure 9 it is assumed that the networks in Figure 9a, 9b and 9c are operational 77%, 11.5%, and 11.5% of the time in a year, respectively. Hence if C_{H_1} is 0.1, then $C_{H_1'}$ and $C_{H_1''}$ are 0.015 each.

The cost of exchanger area at a site will be illustrated with the help of an example. At the site 2, the exchangers of different sizes required in Figure 9 are A_2, A_2', and $A_{2''}$. Assume that A_2' is smallest and $A_{2''}$ the largest area.

The cost of exchanger area at site 2 is defined as

$$35[(A_2')^{0.6} + (A_2 - A_2')^{0.6} + (A_{2''} - A_2)^{0.6}]$$

This manner of costing areas in doing reliability analysis appears to be a good formulation of the real system. If, because of

TABLE 4

ADDITIONAL USER SPECIFICATION FOR THE PROBLEM IN FIGURE 9

Stream	H_1	H_1'	H_1''	UC"
Flow Rate	*	*	*	*
Lower Bound on Flow	50,000	50,000	50,000	0
Upper Bound on Flow	200,000	200,000	200,000	100,000
Inlet Temp	600°	600°	600°	100°
Outlet Temp	*	*	*	150°
Lower Bound on Outlet Temp	190°	190°	190°	150°
Upper Bound on Outlet Temp	600°	600°	600°	150°
Cost Coefficient	0.10	0.015	0.015	0.008

*Unspecified.

Let U_i represent the heat transfer coefficient for exchanger i

$U_1 = U_1'' = 700$

$U_2 = U_2' = U_2'' = 477.27$

$U_3 = U_3' = U_3'' = 562.5$

$U_4 = U_4' = U_4'' = 700$

$U_5'' = 225$

some departure from the normal mode, more exchanger area is required at a particular site, then one must pay for the auxiliary exchanger. If the exchanger area required is more for the normal mode, then an auxiliary exchanger is already in effect. The results obtained on optimization of the system in Figure 9 are shown in Table 5.

II.8. Results and Discussion

A typical application of EROS to a heat exchanger network is illustrated by the example in Figure 10, the stream specifications for which are shown in Table 6. Stream I is a flue gas and streams IV and VIII are utility streams. The flow rates for these streams are not defined but are required to be within specified bounds. Some of the streams in the network are also multicomponent and are characterized by a dew and a bubble point. Thus the network is analyzed to ensure that minimum approach temperature violation does not occur inside the exchanger owing to discontinuities at the dew and bubble points. In fact, at the optimal solution for the example in Figure 10, the minimum approach temperature constraint at the bubble point of stream III is operative inside exchanger 5. The program can also evaluate variables that already exist, fully or in part. For example, exchanger 3 was assumed present with an area of 1500 units. The objective function ϕ is defined as

$$\phi = \sum_{\text{streams I, IV, VIII}} \text{cost coefficient} * \text{Flow Rate} + 35 \sum_{i=1 (\neq 7,8)}^{13} (A_i + 10)^{0.6} - 10^{0.6}$$

TABLE 5

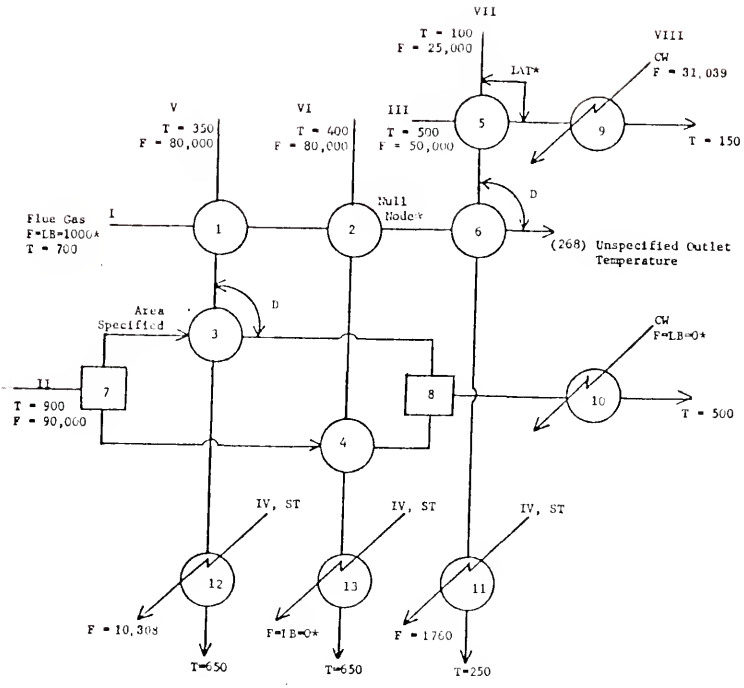
RESULTS FOR THE PROBLEM IN FIGURE 9

Exchanger	Area
1	131.50
2	30.01
3	413.24
4	103.24
$2'$	41.80
$3'$	462.31
$4'$	105.06
$1''$	131.50
$2''$	36.10
$3''$	0.00
$4''$	64.67
$5''$	0.00

$F_{H_1} = 170,341$ $F_{H_1'} = 180,878$ $F_{H_1''} = 50,148$

$T = T' = T'' = 190^{\circ}$

$\phi = \$23451.25/yr.$



ST = STEAM, CW = COOLING WATER
LB = LOWER BOUND
D = MIN. ALLOWABLE APPROACH TEMP.
LAT = INTERNALLY HELD MIN. APPROACH TEMP.
* = ACTIVE CONSTRAINTS AT OPTIMUM

Fig. 10. An Example Exchanger Network

TABLE 6

STREAM SPECIFICATIONS FOR THE EXAMPLE IN FIGURE 10

DESCRIPTION	STREAM							
	I	II	III	IV	V	VI	VII	VIII
FLOW	*	90,000.	50,000.	*	80,000.	80,000	25,000.	*
INLET TEMPERATURE	100.0	900.0	500.0	756.0	350.	400.0	100.0	80.0
OUTLET TEMPERATURE	*	500.0	150.0	756.0	650.	650.0	250.0	130.0
INLET VAPOR FRACTION	0.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0
OUTLET VAPOR FRACTION	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0
LIQUID HEAT CAPACITY	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
VAPOR HEAT CAPACITY	1.0	1.0	*	1.0	1.0	1.0	1.0	1.0
HEAT OF VAPORIZATION	100.0	100.0	75.	768.0	100.0	100.0	200.0	100.0
DEW POINT	800.0	400.0	500.0	756.0	700.0	900.0	250.0	400.0
BUBBLE POINT	800.0	200.0	250.0	756.0	700.0	900.0	250.0	400.0
LIQUID PHASE HTC	1500.0	300.0	300.0	1500.0	300.0	300.0	300.0	300.0
VAPOR PHASE HTC	1500.0	300.0	300.0	1500.0	300.0	300.0	300.0	300.0
TWO PHASE HTC	1500.0	300.0	300.0	1500.	300.0	300.0	300.0	300.0
COST COEFFICIENT	10.0	0.0	0.0	8.5	0.0	0.0	0.0	1.0
LOWER BOUND ON FLOW	1000.	100.0	50,000.	0	80,000.	80,000.	25,000.	0.0
UPPER BOUND ON FLOW	90,000.	90,000.	50,000.	90,000.	80,000.	80,000.	25,000.	90,000.

*Unspecified

HTC - Heat Transfer Coefficient

Minimum Allowable Approach Temperature = 18°

where A_I is the area associated with exchanger i. In the calculation of cost associated with an exchanger, the relation

$$\text{cost} = 35[(A + 10)^{0.6} - 10^{0.6}] \qquad (1)$$

is used in preference to

$$\text{cost} = 35\ A^{0.6} \qquad (2)$$

because whenever the area of an exchanger currently at zero value, is increased, the cost for the exchanger as calculated from (2) increases abnormally compared to the change in cost for the rest of the system. The modification as shown in relation (1) dampens this ill-behaved effect.

There are 8 decision variables for this problem and the optimum results after 328 iterations from an infeasible starting point. The stopping criterion is a 1×10^{-5} difference between the worst and best objective function values in the current set of points retained by the complex algorithm. The value of $\emptyset$ at the optimum is \$152,109/yr.

Results for 10 examples are shown in Table 7. In all the examples the feasible point results in very few iterations. It may be observed that the time required for the rewriting of solution procedures after finding a feasible point is relatively small as compared to the time taken for function evaluations during optimization. The maximum ratio of these two times occur in Example 6, but it is still less than 1/3. This observation indicates that the penalty paid for rewriting solution procedures whenever constraint violations occur is indeed very small.

TABLE 7

RESULTS

Example	1	2	3	4	5	6	7*	8**	9	10
Process Streams	3	4	3	4	2	2	7	5	5	6
Utility Stream	2	1	2	3	2	3	0	3	3	5
Exchangers	4	4	5	7	5	5	9	11	13	14
Decision Variables	2	3	3	4	4	5	6	8	9	13
Iterations to Feasible Point	4	2	2	13	4	9	18	8	8	****
New Solution Procedures After Feasible Time (seconds)	0 0.00	0 0.00	3 0.22	0 0.00	0 0.00	6 3.39	4 1.88	7 7.32	4 3.14	12 11.08
Iterations After Feasible Time (seconds)	35 3.43	59 6.31	72 4.51	108 38.28	149 16.89	107 10.85	706 163.25	320 71.62	119 17.13	819 306
Total Time (seconds)	7.31	8.24	6.48	48.12	36.63	27.82	180.57	108.60	23.49	477

* From Takamatsu, Hashimoto and Ohno (1970).

** Example illustrated in Figure 10.

**** Feasible starting point provided as an input.

The figures shown for time in Table 7 are those required on IBM 360/67. The cost per second of CPU time is about 1.4 cents and the longest run (Example 10) costs $8.56 for complete execution while Example 1 costs $0.40. The size limitations are 50 process stream, 25 nodes and 150 stream segments in the current version of the program. The program is fairly well tested and gave satisfactory results when used for sixteen different problems set up by students in a recent design course.

CHAPTER III
LOCAL AND GLOBAL OPTIMA

III.1. Statement of the Problem

In the course of optimizing a chemical engineering system using a conventional minimum seeking algorithm, one should ask and then attempt to discover if the solution found is indeed a global one. If one is willing to try, then a common strategy is to restart the optimization at several different initial points, and, if all or most lead to a single best point, that point is conjectured to be the global optimum.

A second common strategy is to use one's intuition to claim that only one optimum is likely. This strategy is particularly dangerous when the optimum is at the boundary of the search region where portions of the system effectively disappear because the flow through them is zero. Often one models the capital cost of a process unit by an equation of the form

$$\text{cost} = C_1 \ (\text{Throughput})^{0.6}$$

This form is not convex and is particularly troublesome at zero throughput, where the slope is infinite for cost versus throughput. If a unit becomes zero in size, then a small positive perturbation in its throughput has an apparent positive infinite effect on cost, an effect usually large enough to trap the optimization algorithm. One can of course (and should) modify the cost equation to reduce this problem.

These strategies may help but still do not guarantee that one has found the global optimum. The purpose of the work leading to this paper was to produce a reliable method to determine if an optimal solution is either a global or a local optimal solution. It was hoped that the technique would be computationally effective, in particular for heat exchanger networks, a class of problems which commonly displays local optima.

The optimization problem relating to a heat-exchanger network is to minimize the annualized cost, $\emptyset$, of heat-exchangers and utilities subject to the approach temperature inequality constraints and the heat and material balances representing equality constraints.

The simple network in Figure 11 (due to Grossman and Sargent (1977)) has only one degree of freedom and the network cost can be explored by choosing different values for the temperature T_3. Figure 12 represents the plot of costs, $\emptyset$, vs. T_3. At both the points A and B, the Kuhn-Tucker optimality conditions are satisfied as any slight perturbation away from each of these points results in an increase in $\emptyset$. However, the point B clearly represents a local optimum. It is quite likely therefore that an optimization algorithm will not find the global optimum.

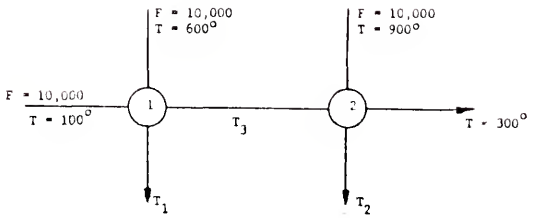
III.2. The Lower Bound

To permit decomposition of a system structure, the primal or overall system optimization problem may be written as

$$\text{Minimize } \emptyset = f(\underline{q}) = \sum_{j=1}^{n} f_j(q_j)$$

$$\text{s.t. } g_i = x_i - t_i(q_i) = 0 \quad i \varepsilon O_{(j)} \quad j=1,2,\ldots n$$

$$q_j \varepsilon S_j \quad j=1,2,\ldots n \qquad \text{(P1)}$$



$C_p = 1$ $U_1 = U_2 = 200$

Area = $Q/u\ \Delta T_{lm}$ Cost = $35 \times \text{Area}^{0.6}$

Cost_i refers to exchanger i.

Aim: To minimize the objective function ϕ, where $\phi = \text{Cost}_1 + \text{Cost}_2$

Fig. 11. Example Problem 1

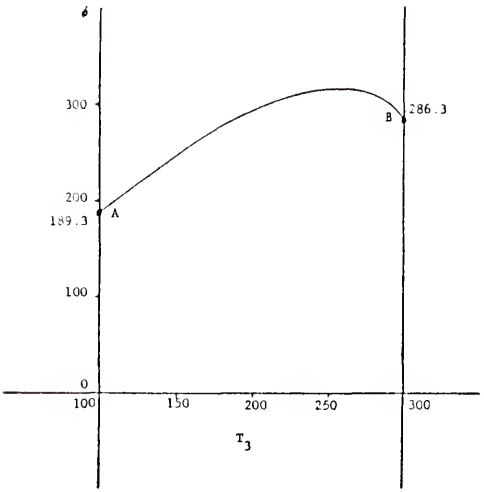


Fig. 12. The Behavior of the Objective Function for Example Problem 1

(It is important to note that the constraints defining S_j must be complete enough to guarantee the boundedness of each subproblem j.)

This representation of a system, such as the one in Figure 13, is based on Lasdon (1970), but a procedure for equality rather than inequality constraints is stressed as shown in McGalliard and Westerberg (1972). This stress follows from an interest in decomposed system structures, as against, say, allocation of resources.

The problem is assumed to have an optimal feasible solution. The Lagrange function for problem (P1) is

$$L = \emptyset - \sum_{j=1}^{n} \sum_{i \in O_{(j)}} \lambda_i [x_i - t_i(q_i)]$$

which may be rewritten as

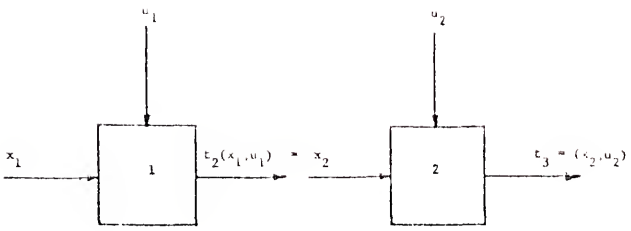
$$L = \sum_{j=1}^{n} [f_j(q_j) + \sum_{i \in O_{(j)}} \lambda_i t_i(q_i) - \sum_{i \in I_{(j)}} \lambda_i x_i] = \sum_{j=1}^{n} \hat{f}_j(q_j, \underline{\lambda})$$

This Lagrange function is separable in $\underline{q}$, and a Lagrange problem may be defined which is equivalent to the following set of subproblems, one for each subsystem in the original system:

$$\underset{q_j \in S_j}{\text{Minimize}} \ \hat{f}_j(q_j, \lambda) \qquad j=1,2,\ldots n \tag{P2}$$

The solution to the Lagrange problem equals the sum of the solutions of (P2), i.e.,

$$\sum_{j=1}^{n} \hat{f}_j^*(\underline{\lambda})$$



u_1 and u_2 are decision variables

Fig. 13. Staged Processes

A dual function can now be defined as

$$h(\underline{\lambda}) = \sum_{j=1}^{n} \hat{f}_j^*(\underline{\lambda})$$

The geometric significance of the dual $h(\underline{\lambda})$ is illustrated in Figure 14 for a simple system like the one shown in Figure 13. The dual $h(\hat{\lambda})$ is the intercept of the line with slope $\hat{\lambda}$. Note that this line is a supporting hyperplane at the point $(\emptyset^o, g^o)$. The geometric significance of the dual has been treated in Lasdon (1970). The value of the dual is always less than or equal to the value of the primal optimum. Hence $h(\underline{\lambda})$ can be considered a lower bound on the global optimum.

III.3. The Upper Bound on the Lower (Dual) Bound

Theorem 1: If in the space of $\emptyset$ vs. $\underline{g}$, where $\underline{g}$ is n-dimensional, a polytope (Geoffrion (1969)) is formed in the hyperplane passing through n+1 support points, the value of $\emptyset$ at the intersection of this polytope with the axis $\underline{g} = 0$ provides an upper bound on the lower bound if the point of intersection is contained inside or at the boundaries of the polytope.

Discussion: Figure 15 illustrates the ideas underlying this theorem. g^1 and g^2 are support points to the graph of $\emptyset$ vs. g. The line connecting g^1 to g^2 is a polytope formed in the hyperplane passing through these n+1 = 2 support points. This polytope or line between g^1 and g^2 intersects the vertical axis g = 0, and thus the

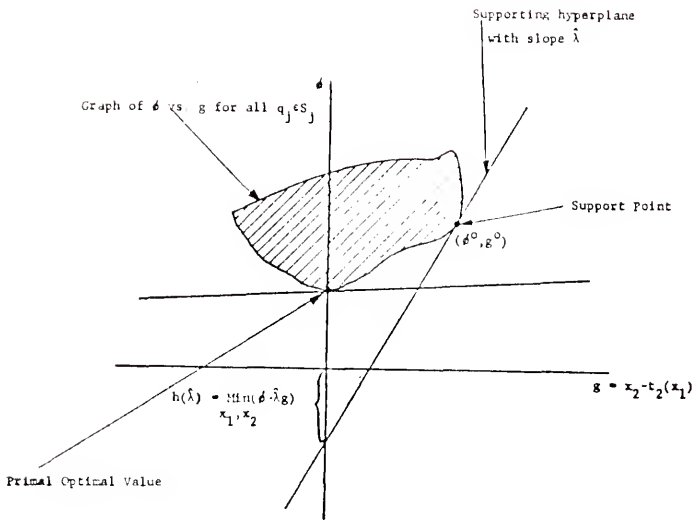


Fig. 14. Geometric Significance of the Dual

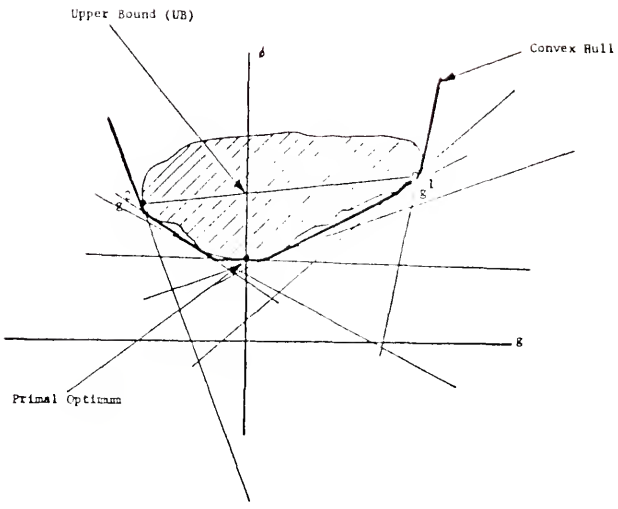


Fig. 15. Geometric Significance of the Upper Bound

theorem says this point of intersection represents an upper bound on the lower bound.

Proof: All the support points of a graph (such as g^1 and g^2) are contained on the surface of a convex hull formed by the support hyperplanes for this graph. Assuming that the polytope intersects the axis $\underline{g} = \underline{0}$ in its interior or at its boundaries, a hyperplane intersecting the axis $\underline{g} = \underline{0}$ above this point must intersect a part of the graph, that is, it cannot be a support hyperplane. Thus every support hyperplane must intersect the axis $\underline{g} = \underline{0}$ on or below the point the polytope intersects this axis. Consequently the dual cannot exceed this value of $\emptyset$ (=UB) and UB is an upper bound to the dual.

III.4. Finding the Best Upper Bound

Given support points $(\emptyset^i, \underline{g}^i)$ i=1,2,...p,p≥n+1, the problem is to determine the following.

(a) Find, if possible, a set of n+1 points that yields an upper bound. Posed differently, the problem is to find n+1 points such that the polytope formed by them intersects the axis $\underline{g} = \underline{0}$.

(b) If several sets of n+1 points qualify to provide an upper bound, find the set that yields the lowest value for the upper bound.

Any point inside or at the boundaries of the convex polytope, specifically the point $\underline{g} = \underline{0}$, can be obtained by a convex combination of (n+1) points forming the polytope. This result has been stated and applied in Director et al. (1978). Thus problem (a) can be formulated as a feasible point problem in linear programming. If a feasible solution exists (no artificial variables present in the solution at a nonzero level), then indeed the set of n+1 points does provide an upper bound.

In order to solve problem (b), if a "price" could be associated with each point such that the objective function reflects the value of the upper bound corresponding to a set of n+1 points, then it too is a linear programming problem. The solution to it, if one exists, would yield solutions to both problems (a) and (b) simultaneously. This "pricing" is indeed possible.

For every point on the polytope formed by n+1 points, $\emptyset$ can be represented by the linear relationship

$$\emptyset = \underline{g}^T \underline{a} + b \tag{1}$$

where $\underline{a}$ and b are constants. The value of $\emptyset$ at the intersection with the axis $\underline{g} = 0$ is

$$\emptyset = b \tag{2}$$

then

$$UB = b \tag{3}$$

assuming that $\underline{g} = \underline{0}$ lies inside or at the boundaries.

For n+1 points, equation (1) may be rewritten compactly as

$$\emptyset = \begin{bmatrix} \emptyset^1 \\ \vdots \\ \emptyset^{n+1} \end{bmatrix} = \begin{bmatrix} \underline{g}^{1T} \\ \vdots \\ \underline{g}^{n+1T} \end{bmatrix} \underline{a} + b \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \tag{4}$$

Let $\underline{\alpha}^T = [\alpha^1, \ldots \alpha^{n+1}]$ such that $\sum_{i=1}^{n+1} \alpha^i = 1$, $[\underline{g}^1, \underline{g}^2, \ldots \underline{g}^{n+1}]\ \underline{\alpha} = 0$, and $\alpha^i \geq$ for all i. Premultiplying equation (4) with $\underline{\alpha}^T$ gives

$$\underline{\alpha}^T \underline{\emptyset} = b \tag{5}$$

and from equation (3)

$$UB = \underline{\alpha}^T \underline{\emptyset} \tag{6}$$

Thus the "price" associated with each point $(\emptyset^i, \underline{g}^i)$ is $\emptyset^i$.

The linear programming formulation to solve both problems (a) and (b) simultaneously is as follows:

Find $\underline{\alpha}^T = [\alpha^1, \ldots \alpha^P]$ to

$$\text{Max } Z = -[\emptyset^1, \ldots \emptyset^P] \begin{bmatrix} \alpha^1 \\ \vdots \\ \alpha^P \end{bmatrix}$$

$$\text{s.t. } [\underline{g}^1, \ldots \underline{g}^P] \begin{bmatrix} \alpha^1 \\ \vdots \\ \alpha^P \end{bmatrix} = \underline{0}$$

$$\alpha^1 + \alpha^2 + \ldots + \alpha^P = 1$$

and $\alpha^1, \ldots, \alpha^P, \geq 0$. The upper bound UB = - Z.

The solution will contain (n+1) vectors in the basis. If a feasible solution does not exist, artificial variables will be present in the solution at a positive level and no upper bound will exist because the set of points in hand so far $(\underline{g}^1, \underline{g}^2, \ldots \underline{g}^P)$ do not surround the origin.

III.5. Improvement of the Upper Bound

Let N be the set of the current n+1 points forming the polytope P, $\hat{H}$ the hyperplane passing through these points with the slope

$\underline{\lambda}$, and UB the value of the upper bound. The aim now is to find an improvement on UB.

If a new point is introduced, it follows from the simplex method that, since the problem is bounded, a feasible solution and hence a new upper bound can be obtained by replacing one of the points in N by the new point. The question now is how can the new point be found so that the new upper bound (UB') is an improvement on UB, i.e., UB' < UB.

Once again, the geometry of the problem and the theory of linear programming can be used advantageously. Assume for the moment that the solution to the linear programming problem obtained from the points in N is not degenerate. If a support point is determined for slope $\underline{\lambda}$, this new point must lie on or below $\hat{H}$. This result is obvious as a hyperplane with slope $\underline{\lambda}$ cannot be a support hyperplane above $\hat{H}$. If the coordinates of the new point are $(\emptyset^{new}, \underline{g}^{new})$, and the value of $\emptyset$ on $\hat{H}$ at $\underline{g} = \underline{g}^{new}$ is $\emptyset^{\hat{H}}$, then

$$\emptyset^{\hat{H}} \geq \emptyset^{new}$$

$\emptyset^{\hat{H}} = \emptyset^{new}$ implies that no further improvement can be obtained on the upper bound. Thus if $\emptyset^{\hat{H}}$ is greater than $\emptyset^{new}$, it follows that UB' will be less than UB if the new point replaces one of the points in N by the simplex method. This result can be demonstrated as follows:

Let the points in N be $(\emptyset^1, \underline{g}^1), \ldots (\emptyset^{n+1}, \underline{g}^{n+1})$. The constraints are

$$(\underline{g}^1, \underline{g}^2, \ldots \underline{g}^{n+1})\ \underline{\alpha} = \underline{0} \qquad (7)$$

$$\alpha^1 + \alpha^2 \ldots + \alpha^{n+1} = 1 \tag{8}$$

Combining relations (7) and (8)

$$\hat{\underline{\underline{G}}} \ \underline{\alpha} = \begin{bmatrix} \underline{0} \\ \underline{1} \end{bmatrix} \tag{9}$$

The new point can be represented as a linear combination of the points in the basis, that is,

$$\hat{\underline{g}}^{new} = \begin{bmatrix} \underline{g}^{new} \\ \underline{1} \end{bmatrix} = \hat{\underline{\underline{G}}} \ \underline{y}^{new} \tag{10}$$

The $n+1^{st}$ element of $\hat{\underline{g}}^{new}$ gives

$$1 = \sum_{i=1}^{n+1} y_i^{new} \tag{11}$$

Points on $\hat{H}$ may be expressed by equation (1)

$$\emptyset = [\underline{g}^T, 1] \begin{bmatrix} \underline{a} \\ \underline{0} \end{bmatrix} + b$$

For the (n+1) points in the basis, it follows from equation (4) that

$$\emptyset = \hat{\underline{\underline{G}}}^T \begin{bmatrix} \underline{a} \\ \underline{0} \end{bmatrix} + b \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

Premultiplying with $\underline{y}^{new^T}$,

$$\underline{y}^{new^T} \emptyset = \underline{y}^{new^T} \hat{\underline{\underline{G}}}^T \begin{bmatrix} \underline{a} \\ \underline{0} \end{bmatrix} + b \sum_{i=1}^{n} y_i^{new} \tag{12}$$

Using equations (10) and (11)

$$\underline{y}^{new^T} \underline{\emptyset} = \begin{bmatrix} \underline{g}^{new^T}, 1 \end{bmatrix} \begin{bmatrix} \underline{a} \\ \underline{0} \end{bmatrix} + b \tag{13}$$

However, the right hand side is a point on $\hat{H}$ at $\underline{g}^{new}$ and thus

$$\underline{y}^{new^T} \underline{\emptyset} = \emptyset^{\hat{H}} \tag{14}$$

In the simplex method, the criterion for an improvement in the objective function by replacing a point in the basis by a new one is

$$\underline{y}^{new^T} \underline{\emptyset} - \emptyset^{new} = \emptyset^{\hat{H}} - \emptyset^{new} > 0 \quad \text{(see Hadley (1963))} .$$

Since $\emptyset^{\hat{H}} - \emptyset^{new}$ is greater than zero

$$UB' < UB$$

If the linear programming solution corresponding to the points in N is degenerate, that is, the point $\underline{g} = \underline{0}$ lies on one of the boundaries of P, a difficulty arises. In this case, there is no unique hyperplane $\hat{H}$ and consequently there may be some degrees of freedom available in calculating $\underline{\lambda}$. There is then no guarantee that the new point will bring about an improvement in the upper bound. In such a case, several new points may have to be evaluated before an upper bound with a lower value is obtained. Degenerate linear programming problem solutions occur in Example Problem 3.

III.6. Procedure to Investigate Global Optimality

The basic idea of this approach is that the value for the global optimum must lie between or at the boundary of the upper and lower bounds. The interval between the bounds is generally decreased at every iteration by the improvement on the upper bound and possibly an improvement on the lower bound. If at some point during this

procedure, an optimum is found to possess a value significantly greater than the upper bound, an inference can be made that the optimum is local and the procedure terminated. On the other hand, if a value for the optimum is very close to the lower bound, it may be inferred that the optimum is global and the process terminated.

A problem arises when there is a dual gap in the optimization problem. It is possible that the global optimum may lie above the upper bound at some iteration. However, the optimum is not rejected as a local optimum if it lies within a certain interval above the upper bound, say 2% of the value of the optimum. Previous experience indicates that this modification is adequate for the type of problems considered in this study. This behavior can be observed in Example Problem 2.

III.7. Algorithm

To investigate a primal optimal solution with an objective function $\emptyset^*$.

Step 1 Guess n+1 different $\underline{\lambda}$'s ($\underline{\lambda}^j$ where j=1 to n+1). Attempt to select these $\underline{\lambda}^j$ to obtain points $\underline{g}$ which surround the origin.

Step 2 For each $\underline{\lambda}^j$, evaluate the dual $h(\underline{\lambda}^j)$, the support point $\underline{g}^j$, and the objective function $\emptyset^j$. Let p = the total number of support points (n+1) in this case). Find LB where LB = Max ($h(\underline{\lambda}^j)$, j=1 to p).

Step 3 If $\emptyset^* \leq$ LB + ε (ε is a small positive quantity, say .02 $\emptyset^*$), stop. $\emptyset^*$ is the global optimum. Else go to Step 4.

Step 4 Formulate an L.P. problem with all the available support points and solve for the optimal objective function (-UB) and the corresponding basis vector. If a feasible solution does not exist, go to Step 7. If $\emptyset^* \geq$ UB + ε, stop. The given solution is a local optimum. If the L.P. solution is degenerate, go to Step 6.

Step 5 In the L.P. solution, let the vectors in the basis be $\underline{g}^1, \underline{g}^2, \underline{g}^3, \ldots \underline{g}^{n+1}$. Set up n simultaneous equations in unknowns $\underline{\lambda}^{p+1}$ as follows

$$\begin{aligned}
(\underline{g}^2 - \underline{g}^1)^T \underline{\lambda}^{p+1} &= \phi^2 - \phi^1 \\
(\underline{g}^3 - \underline{g}^1)^T \underline{\lambda}^{p+1} &= \phi^3 - \phi^1 \\
(\underline{g}^4 - \underline{g}^1)^T \underline{\lambda}^{p+1} &= \phi^4 - \phi^1 \\
(\underline{g}^{n+1} - \underline{g}^1)^T \underline{\lambda}^{p+1} &= \phi^{n+1} - \phi^1
\end{aligned} \tag{15}$$

Solve for $\underline{\lambda}^{p+1}$. Find the dual $h(\underline{\lambda}^{p+1})$, the support point $\underline{g}^{p+1}$, and the objective function ϕ^{p+1}. Set $p = p + 1$. Find LB where

$$LB = \text{Max } (h(\underline{\lambda}^j), \quad j=1,2,\ldots p)$$

Go to Step 3.

Step 6 Select all vectors from the basis which are present in the L.P. solution in Step 4 at a nonzero level. Let these be points $\underline{g}^1, \underline{g}^2, \ldots \underline{g}^m$. Since only m (< n+1) such points exist, the $\underline{\lambda}^{p+1}$ to be chosen can come anywhere from within an n + 1 - m dimensional subspace. Either initiate (if first time through this step with above nonzero basis vectors) or continue a systematic search over the subspace to find the next $\underline{\lambda}$'s to use. A $\underline{\lambda}$ from the appropriate subspace is found by having it satisfy

$$\begin{aligned}
(\underline{g}^2 - \underline{g}^1)^T \underline{\lambda}^{p+1} &= \phi^2 - \phi^1 \\
(\underline{g}^3 - \underline{g}^1)^T \underline{\lambda}^{p+1} &= \phi^3 - \phi^1 \\
(\underline{g}^m - \underline{g}^1)^T \underline{\lambda}^{p+1} &= \phi^m - \phi^1
\end{aligned} \tag{16}$$

plus any additional n + 1 - m independent specifications. Go to Step 3.

Step 7 Select a set of $\underline{\lambda}^{p+1}$ to find a new support point $\underline{g}^{p+1}$, the dual $h(\underline{\lambda}^{p+1})$, and the objective function ϕ^{p+1}. Set $p = p + 1$. Let LB = Max $(h(\underline{\lambda}^j)$, where j=1 to p). Go to Step 3. (Note this step is actually the same as Step 6 but with no constraints of form (16) being written, i.e., one should search systematically over the entire space for the next $\underline{\lambda}$'s.)

III.8. Examples

Example Problem 1

Figure 11 represents the Example Problem 1, and its subsystems are shown in Figure 16.

$$h(\underline{\lambda}) = \underset{y_1}{\text{Min}} \ (\text{cost}_1 + \lambda_1 y_1) + \underset{x_1}{\text{Min}} \ (\text{cost}_2 - \lambda_1 x_2)$$

where λ_1 is the price associated with the variable T_3.

$$g_1 = (x_2 - y_1)$$

On optimization of the original system the two possible solutions are

Case: 1 $T_1 = 400°$ $T_2 = 900°$ $T_3 = 300°$ $\phi^* = 286.93$

Case: 2 $T_1 = 600°$ $T_2 = 700°$ $T_3 = 100°$ $\phi^* = 189.3$

Apply the algorithm to Case 1, $\phi^* = 286.93$

Step 1 $\lambda_1^1 = -4.14$; $\lambda_1^2 = 4.14$ and $p = 2$ points.

Step 2 $g_1^1 = -200$ $h(\underline{\lambda}_1^1) = -224.6$ $\phi^1 = 476.2$

$g_1^2 = +200$ $h(\underline{\lambda}_1^2) = -882.0$ $\phi^2 = 0$

LB = -224.6

Step 3 $\phi^* > \text{LB} + \varepsilon$. (Let $\varepsilon = 0.02\ \phi^* = 5.74$.)

Step 4 UB = 238.1

$\phi^* > \text{UB} + \varepsilon$

Hence the solution in Case 1 is a local optimum.

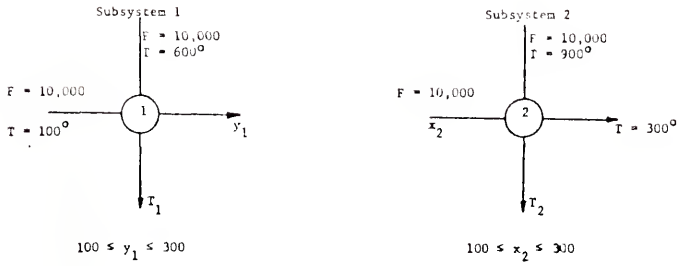


Fig. 16. Subsystems for Example Problem 1

Apply the algorithm to Case 2, $\phi^* = 189.3$ Steps 1, 2 and 3 are the same as in Case 1.

Step 4 UB = 238.1

$\phi^* < UB + \varepsilon$

Step 5 $\lambda_1^3 = -1.19$

$h(\underline{\lambda}_1^3) = 189.3 \quad , \quad g_1^3 = 0 \quad , \quad \phi^3 = 189.3$

$p = 3$

$LB = 189.3$

Step 3 $\phi^* = 189.3 < LB + \varepsilon$

Hence the solution in Case 2 is a global optimum.

Example Problem 2

Figure 17 represents this problem and the subsystems are shown in Figure 18.

$$h(\lambda) = \underset{y_{11}}{\text{Min}} \ (\text{Area}_1 + \lambda_1 y_{11}) + \underset{y_{12}, x_{22}}{\text{Min}} \ (\text{Area}_2 + \lambda_2 y_{12} - \lambda_1 x_{22})$$

$$+ \underset{x_{23}}{\text{Min}} \ (\text{Area}_3 - \lambda_2 x_{23})$$

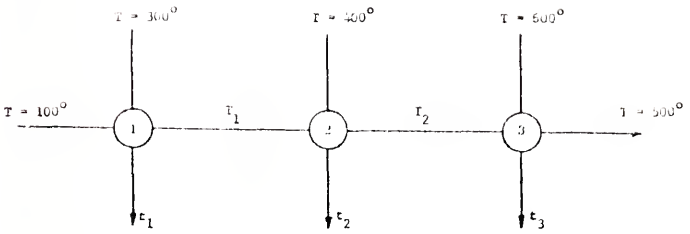
λ_1 is the price associated with T_1 and λ_2 is the price associated with T_2.

$$g_1 = (x_{22} - y_{11}) \quad \text{and} \quad g_2 = (x_{23} - y_{12})$$

On optimization

$T_1 = 181.9°$ $T_2 = 295°$ $t_1 = 218.1°$ $t_2 = 286.4°$

$t_3 = 395.5°$ and $\phi^* = 7049.$



$FC_p = 10^5$ for all the streams

$U_1 = 120$ $U_2 = 80$ $U_3 = 40$

$Area_1$ refers to exchanger 1

$Area = Q/U \Delta T_{lm}$, $\phi = Area_1 + Area_2 + Area_3$

Aim: To minimize ϕ

Fig. 17. Example Problem 2

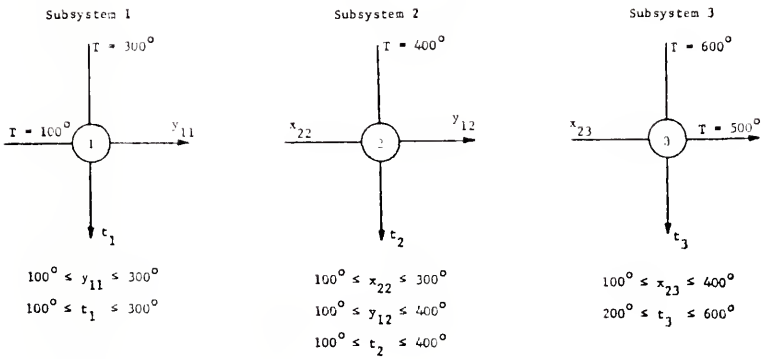


Fig. 18. Subsystems for Example Problem 2

Application of the Algorithm: $\emptyset^* = 7049$

Step 1 $\underline{\lambda}^1 = \begin{bmatrix} -10 \\ 0 \end{bmatrix}$ $\underline{\lambda}^2 = \begin{bmatrix} 0 \\ 10 \end{bmatrix}$ $\underline{\lambda}^3 = \begin{bmatrix} -15 \\ -30 \end{bmatrix}$ $p = 3$

Step 2 $\underline{g}^1 = \begin{bmatrix} 200 \\ 100 \end{bmatrix}$ $\underline{g}^2 = \begin{bmatrix} 0 \\ 300 \end{bmatrix}$ $\underline{g}^3 = \begin{bmatrix} -94.6 \\ -188 \end{bmatrix}$

$h(\underline{\lambda}^1) = 500$ $h(\underline{\lambda}^2) = -500$ $h(\underline{\lambda}^3) = 5787$

$\emptyset^1 = 2500$ $\emptyset^2 = 2500$ $\emptyset^3 = 12{,}846$

$LB = 5787$

Step 3 $\emptyset^* > LB + \varepsilon$ Let $\varepsilon = 140$

Step 4 $UB = 9001.46$

$\emptyset^* < UB + \varepsilon$

Step 5 $\underline{\lambda}^4 = \begin{bmatrix} -21.67 \\ -21.67 \end{bmatrix}$ $\underline{g}^4 = \begin{bmatrix} -112.3 \\ 132 \end{bmatrix}$

$h(\underline{\lambda}^4) = 5585$ $\emptyset^4 = 5158$ $p = 4$

$LB = 5787$

Step 3 $\emptyset^* > LB + \varepsilon$

Step 4 $UB = 7173.34$

$\emptyset^* < UB + \varepsilon$

Step 5 The points in the basis are $\underline{g}^1$, $\underline{g}^3$ and $\underline{g}^4$.

$\underline{\lambda}^5 = \begin{bmatrix} -11.04 \\ -24.65 \end{bmatrix}$ $\underline{g}^5 = \begin{bmatrix} 122.87 \\ 71 \end{bmatrix}$

$h(\lambda^5) = 6640.3$ $\emptyset^5 = 3533.67$ $p = 5$

$LB = 6640.3$

Step 3 $\emptyset^* > LB + \varepsilon$

Step 4 UB = 6928.94

$\phi^* < UB + \varepsilon$

Step 5 The points in the basis are $\underline{g}^3$, $\underline{g}^4$ and $\underline{g}^5$.

$$\underline{\lambda}^6 = \begin{vmatrix} -13.34 \\ -24.76 \end{vmatrix} \qquad \underline{g}^6 = \begin{vmatrix} 111.79 \\ 71 \end{vmatrix}$$

$h(\underline{\lambda}^6) = 6916.26 \quad \phi^6 = 3668.11 \quad p = 6$

$LB = 6916.26$

Step 3 $\phi^* < LB + \varepsilon$

Hence the solution represents a global optimum.

Example Problem 3

Figure 2 represents this problem and the subsystems are shown in Figure 19.

y_{11}, x_{22} and price λ_1 are associated with F_2

y_{12}, x_{24} and price λ_2 are associated with $(F_4 \times T_4)$

y_{22}, x_{25} and price λ_3 are associated with T_{12}

y_{13}, x_{26} and price λ_4 are associated with T_{15}

$$h(\underline{\lambda}) = \underset{x_{22},y_{12},y_{22}}{\text{Min}} \; (Cost_1 - \lambda_1 x_{22} + \lambda_2 y_{12} + \lambda_3 y_{22}) +$$

$$\underset{y_{11},x_{24},y_{13}}{\text{Min}} \; (Cost_2 + \lambda_1 y_{11} - \lambda_2 x_{24} + \lambda_4 y_{13}) +$$

$$\underset{x_{25}}{\text{Min}} \; (Cost_3 + 0.4F_{II} - \lambda_3 x_{25}) +$$

$$\underset{x_{26}}{\text{Min}} \; (Cost_4 + 0.2F_{III} - \lambda_4 x_{26}) +$$

and

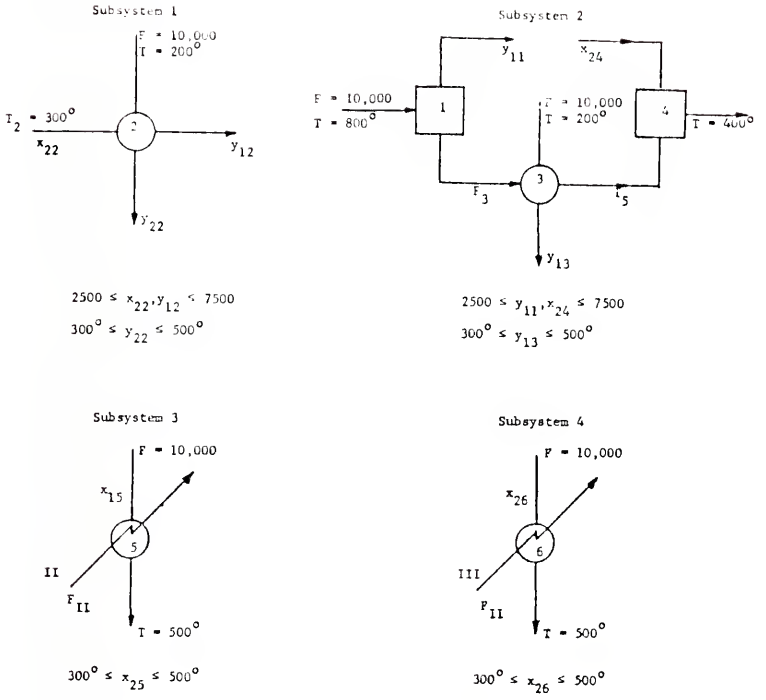


Fig. 19. Subsystems for Example Problem 3

$$g_1 = (x_{22} - y_{11})$$

$$g_2 = (x_{24} - y_{12})$$

$$g_3 = (x_{25} - y_{22})$$

$$g_4 = (x_{26} - y_{13})$$

On optimization there are two possible solutions:

Case: 1 $F_2 = 2500$ $F_3 = 7500$ $T_4 = T_5 = 400°$ $T_{12} = 300°$

$T_{15} = 500°$ $F_{11} = 2500$ $F_{III} = 0$ $\phi^* = 4154$

Case: 2 $F_2 = 7350$ $F_3 = 2650$ $T_{12} = 500°$ $T_{15} = 300°$

$F_{II} = 0$ $F_{III} = 2500$ $T_5 = 423°$ $T_4 = 392°$

$\phi^* = 3538$

Application of the Algorithm to Case 1 $\phi^* = 4154$

Step 1 $\underline{\lambda}^1 = \begin{bmatrix} 0 \\ 0 \\ -10 \\ -10 \end{bmatrix}$ $\underline{\lambda}^2 = \begin{bmatrix} 0 \\ 0.000555 \\ 0 \\ 0 \end{bmatrix}$ $\underline{\lambda}^3 = \begin{bmatrix} 0 \\ 0 \\ -10 \\ 0 \end{bmatrix}$

$\underline{\lambda}^4 = \begin{bmatrix} -0.1 \\ 0.001 \\ -10 \\ -10 \end{bmatrix}$ $\underline{\lambda}^5 = \begin{bmatrix} 0 \\ 0.001 \\ -10 \\ 5 \end{bmatrix}$ $p = 5$

Step 2 $\underline{g}^1 = \begin{bmatrix} 5000 \\ -2,000,000 \\ -200 \\ -200 \end{bmatrix}$ $\underline{g}^2 = \begin{bmatrix} -5000 \\ 2,000,000 \\ 200 \\ 200 \end{bmatrix}$ $\underline{g}^3 = \begin{bmatrix} 2345 \\ -1,876,000 \\ -200 \\ 200 \end{bmatrix}$

$\underline{g}^4 \begin{bmatrix} -5000 \\ 2,000,000 \\ 0 \\ 0 \end{bmatrix}$ $\underline{g}^5 = \begin{bmatrix} -2345 \\ 1,876,000 \\ -200 \\ 200 \end{bmatrix}$

$h(\underline{\lambda}^1) = 2350$ $\quad \phi^1 = 6350$

$h(\underline{\lambda}^2) = 230$ $\quad \phi^2 = 1340$

$h(\underline{\lambda}^3) = 1917$ $\quad \phi^3 = 3917$

$h(\underline{\lambda}^4) = 2118$ $\quad \phi^4 = 4618$ $\quad$ LB = 2350

$h(\underline{\lambda}^5) = 2188$ $\quad \phi^5 = 5064$

Step 3 $\phi^* > LB + \varepsilon$ $\quad$ Let $\varepsilon = 80$

Step 4 UB = 3845

$\phi^* > UB + \varepsilon$

Hence Case 1 corresponds to a local optimum.

Application of the Algorithm to Case 2 $\quad \phi^* = 3538$

Steps 1 to 3 same as in Case 1.

Step 4 UB = 3845

$\phi^* < UB + \varepsilon$

LP solution is degenerate

Step 6 Nonzero points in the basis of LP solution are $\underline{g}^1$ and $\underline{g}^2$.

Let the n+1 points for finding $\underline{\lambda}^6$ be $\underline{g}^1$, $\underline{g}^2$, $\underline{g}^3$, $\underline{g}^4$, and $\underline{g}^5$.

$$\underline{\lambda}^6 = \begin{bmatrix} -0.065 \\ 0.000225 \\ -9.81 \\ -6.58 \end{bmatrix} \qquad \underline{g}^6 = \begin{bmatrix} 0 \\ 0 \\ -200 \\ 200 \end{bmatrix}$$

$h(\underline{\lambda}^6) = 3338$ $\quad \phi = 4037$

p = 6 $\quad$ LB = 3338

Step 3 $\phi^* > LB + \varepsilon$

Step 4 UB = 3845

$\phi^* < UB + \varepsilon$

LP solution is degenerate

Step 6 Nonzero points in the basis of the LP solution are $\underline{g}^1$ and $\underline{g}^2$.

Let the n+1 points for finding $\underline{\lambda}^7$ be $\underline{g}^1$, $\underline{g}^2$, $\underline{g}^4$, $\underline{g}^5$, and $\underline{g}^6$.

$$\underline{\lambda}^7 = \begin{bmatrix} 0.128 \\ 0.000709 \\ -8.68 \\ -7.72 \end{bmatrix} \qquad \underline{g}^7 = \begin{bmatrix} -2345 \\ 4690 \\ 0 \\ 0 \end{bmatrix}$$

$h(\underline{\lambda}^7) = 3534$ $\qquad \phi^7 = 4564$ $\qquad p = 7$

$LB = 3534$

Step 3 $\phi^* < LB + \varepsilon$

Hence Case 2 represents the global optimum.

III.9. Discussion

In the first and the third examples, the local optima are recognized very quickly. The global optimum in each of these examples is confirmed albeit with a greater number of steps. In the second example, a dual gap (global optimum = 7049, upper bound = 6928.9) is present. However, the algorithm proves effective. It may also be noticed in the second example that an improvement results in the upper bound at every iteration. This is guaranteed in the absence of degeneracy. Degenerate solutions occur in the third example and several points may have to be evaluated before obtaining an improvement in the upper bound.

CHAPTER IV
PROCESS SYNTHESIS USING STRUCTURAL PARAMETERS: A PROBLEM WITH INEQUALITY CONSTRAINTS

In this chapter a problem with the structural parameter method for process synthesis shall be discussed. The problem is almost obvious once stated, but it has not been emphasized in the literature. It can seriously affect the expected results. The discussion here does not imply that the method is valueless, it simply stresses that care must be taken to ensure that the method is really useful for a given problem.

A system of interconnected process subsystems can be modeled using structural parameters which are defined by the following equations

$$x_i = \sum_{j=1}^{N} \alpha_{ij} x'_j \qquad i=1,2,3,\ldots N$$

$$0 \le \alpha_{ij} \le 1 \ , \quad \sum_{i=1}^{N} \alpha_{ij} = 1 \qquad (1)$$

where x_i and x'_j are, respectively, the input and output variables of the i-th subsystem, N is the total number of subsystems in the entire system and the parameters α_{ij} are the structural parameters; that is, each is the fraction of the output stream of the j-th subsystem which flows into the i-th subsystem.

By means of structural parameters, a system synthesis problem can be transformed into a non-linear programming problem with continuous

decision variables. This idea has been stated and demonstrated in the studies by Umeda, Hirai and Ichikawa (1972), Ichikawa and Fan (1973), Osakada and Fan (1973), Mishra, Fan and Erickson (1973), and Himmelblau (1975). In order to obtain an optimal structure, redundant subsystems are usually inserted into a "super" structure in which it is hoped that the optimal structure is imbedded. For example, Figure 20 illustrates one use of the method for synthesizing a heat recovery network. The problem data and the stream specifications are shown in Table 8, and the problem is described in the tradition of problems like 4SP1 in Masso and Rudd (1969) and Lee et al. (1970). A decision is to be made on whether or not to use cold stream II to aid in cooling hot stream V. The problem is formulated with potentially redundant exchangers 2, 4 and 5 and the structural parameter α_{26}. α_{26} is the fraction of cold stream II that goes through exchanger 2. The strategy is, in this formulation, to convert the discrete decision of whether or not to use cold stream II to aid in cooling hot stream V into a problem of optimizing over the continuous decision variable α_{26}. To avoid a cost discontinuity caused by introducing an exchanger we must, of course, require the cost of an exchanger to approach zero as its area does.

The notion that we have converted a discrete decision problem to a continuous one is invalid here. It may be observed that hot stream V has sufficient heat content to drive cold stream I to its final temperature. However, the transfer of heat in exchanger I is limited because an inequality constraint in exchanger 2 has to be satisfied. This constraint is the approach temperature requirement

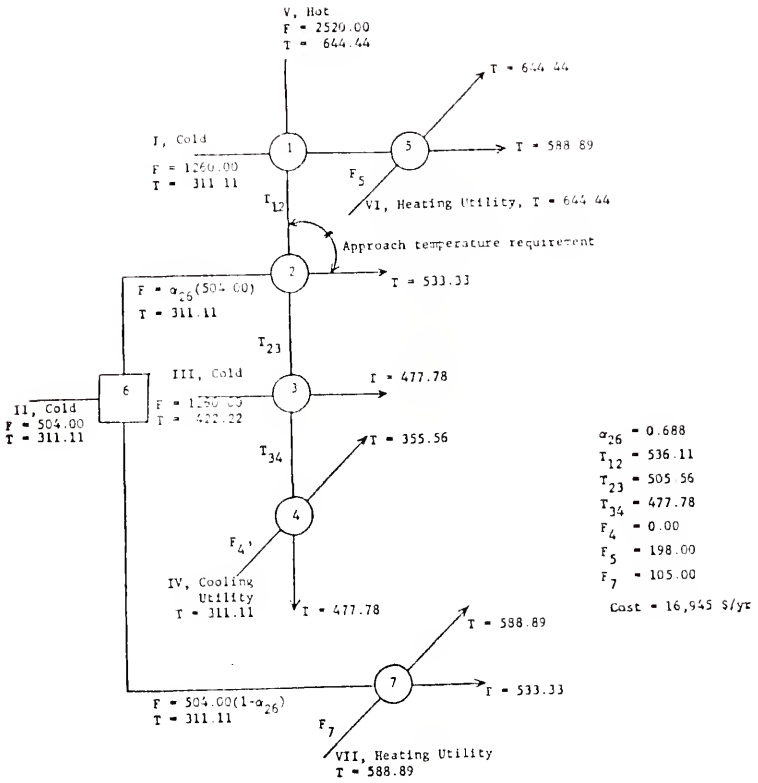


Fig. 20. The Optimal Values of the Parameters

TABLE 8

STREAM SPECIFICATIONS AND PROBLEM DATA

DESCRIPTION	UNITS	STREAM						
		I	II	III	IV	V	VI	VII
Flow	g/s	1260.00	504.00	1260.00	Unknown	2520.00	Unknown	Unknown
Inlet Temperature	K	311.11	311.11	422.22	311.11	644.44	644.44	588.89
Outlet Temperature	K	588.89	533.33	477.78	355.56	477.78	644.44	588.89
Boiling Point	K	755.56	755.56	755.56	373.33	755.56	644.44	588.89
Liquid Heat Capacity	kJ/kg K	4.19	4.19	4.19	4.19	4.19	4.19	4.19
Film Heat Transfer Coefficient	W/m^2 K	1703.49	1703.49	1703.49	1703.49	1703.49	8517.45	8517.45
Cost	$\frac{\$}{g}$	0.00	0.00	0.00	1.10×10^{-7}	0.00	22.05×10^{-7}	5.20×10^{-7}
Heat of Vaporization	kJ/kg	697.80	697.80	697.80	1786.368	697.80	1628.20	1395.60
Inlet Vapor Fraction		0.00	0.00	0.00	0.00	0.00	1.00	1.00
Outlet Vapor Fraction		0.00	0.00	0.00	0.00	0.00	0.00	0.00

Approach Temperature = 2.78°K

Heat exchanger cost equation = $(a A^b)$

where = annual rate of return = 0.1

a = 350

b = 0.6

Equipment down time = 280 hr./yr.

The cost of the network is the cost of utility streams + the cost of the exchangers, in $/yr.

All heat exchangers are assumed to be counter-current

between the inlet hot stream and the outlet cold stream ($T_{12} \geq 533.33° + D$, where D is a minimum allowed approach temperature of, say, 2.78K). By numerical computation one finds that the overall cost can be lowered if α_{26} is increased (see Figure 21). This increase in α_{26} lowers the flows of cooling utilities IV and VII. Ultimately, when the flow of cooling utility IV is zero, an optimal structure is obtained with a cost of \$16,945/yr. At this point $\alpha_{26} = 0.688$.

Note, however, that when the flow of the cold stream through exchanger 2 is zero, and the heat exchanger 2 is totally neglected, a network with a cost of \$6864/yr. is obtained as shown in Figure 22. Thus, a significant discontinuity, which was hoped to have been eliminated, has reappeared. It should be emphasized that the data for the problem and its formulation are important only in that they are plausible and that they help make this point.

The main observation may be summarized as follows. When certain subsystems are rendered redundant during optimization (by the associated flow or a split fraction taking on a value of zero) and if inequality constraints are associated with these subsystems which force constrained behavior elsewhere, discontinuities very likely still exist in the problem. One must still make a discrete decision about whether or not to introduce that subsystem. This observation means that, if the structural parameters are to be used for a synthesis problem, and, if inequality constraints are involved, the problem has to be formulated very carefully, if indeed it can be, to make it a continuous one.

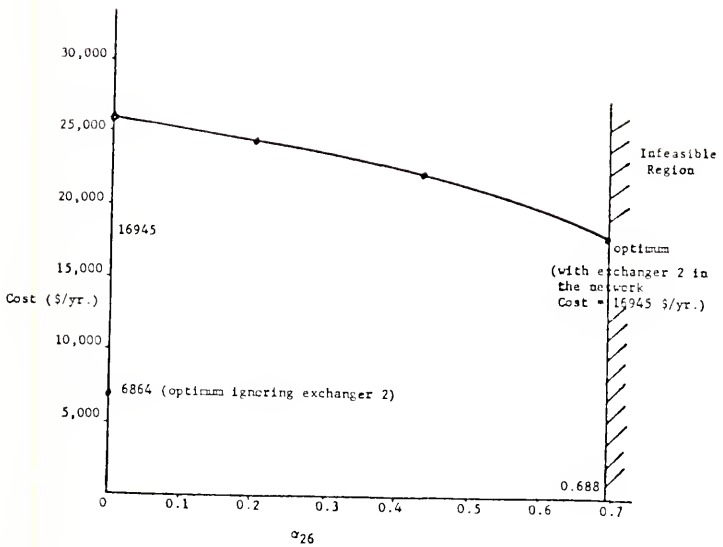


Fig. 21. The Discontinuity in Optimization

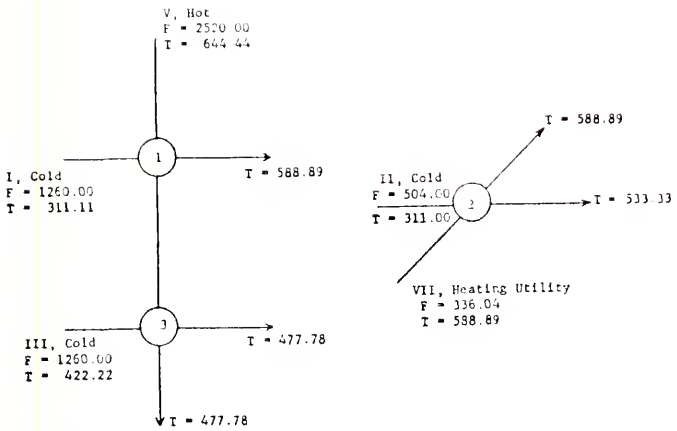


Fig. 22. The Optimal Network When Ignoring the Approach Temperature Requirement at Null Exchanger 2 of Figure 20

CHAPTER V
CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER RESEARCH

The optimization strategy chosen for EROS proves to be efficient. It is the author's belief that the number of steps required for convergence is significantly lowered by rederiving a solution procedure every time a constraint violation occurs, and by the use of 'restriction' (Geoffrion (1970)). A very useful feature in EROS is its ability to find a feasible starting point, if none such is provided by the user. The solution yielded by EROS can account for portions of the network that already exist and for irregularities likely to occur in the process streams. Considering the nature of the problem treated, cost per a typical run of EROS seems small. The use of program may also be extended in carrying out synthesis via structural parameters but the problem must be formulated very carefully as mentioned in Chapter IV. In the results shown in Table 7, a feasible starting point is obtained in very few iterations. A feasible starting point may also be provided by the user as in example 10 of Table 7. As indicated in Chapter II, the time required for the rewriting of solution procedures after finding a feasible point is relatively small as compared to the function evaluations during optimization. Hence, the equation solving approach followed is fully justified.

The practical applicability of a program such as EROS can only be considered complete if a capability of handling power generation and power intensive units is incorporated. Hence there is a need for adding unit models such as turbines and compressors. Hereafter the pressure changes will be important and a pressure variable must be associated with each segment. These changes might also call for a better modeling of a heat-exchanger unit in which the heat-transfer coefficients may no longer be evaluated by simplistic relations. Currently the cooling curves for streams are approximated by three linear regions. A more accurate description could be provided for streams by specifying enthalpy versus temperature information for the range under consideration. A linear interpolation may be assumed between two adjacent points provided in this specification.

In Chapter III an algorithm is presented which can effectively identify if an optimum point is a global optimum or a local one. It is practical as given only when applied to systems of interconnected subsystems, fortunately, a problem form which is common in engineering design. The algorithm develops a lower (dual) bound using the ideas of Brosilow and Lasdon (1965) wherein the total system is decomposed into subsystems, each of which must be globally optimized. If the subsystems are small enough, this requirement is readily satisfied. Frequently the optimal lower (dual) bound is equal to the optimal upper (primal) bound, but it also often lies strictly below because of nonconvexities in the problem. Previous experience on actual engineering problems indicates that this difference, caused by a duality gap, is small, i.e., only a small percent of the system cost.

The algorithm also develops an upper bound to this lower bound. Subject to the tolerances required because of the duality gap, an optimum more than a few percent above this upper bound is declared to be a local optimum, and one at or just above the lower or dual bound is considered to be a global optimum. If no discrimination is possible at the current step, the algorithm provides a means to guarantee improvement of the upper bound for nondegenerate problems, the lower bound usually being improved at the same time. The three example problems demonstrate that the algorithm is surprisingly effective.

Relating to the algorithm presented in Chapter III, a study is recommended to discover the global optimum having discovered that the given primal optimum is local. It would be desirable to predict a value of $\underline{\lambda}$ yielding the global optimum by perceiving a trend in the value of $\underline{\lambda}$ from the information gathered.

APPENDICES

APPENDIX A
DESCRIPTION OF THE INPUT AND OUTPUT FEATURES

DATA INPUT FOR THE PROGRAM EROS

Card	Variable	Card Format	Description
First Card	FL	4F8.1	Scaling factor for Flow Recommended FL = Minimum Flow
	T		Scaling factor for Temperature Recommended T = Minimum Temperature
	A		Scaling factor for split fraction Recommended A = 0.2
	SC		Scaling factor for slack variables Recommended SC = Minimum Temperature
Next Card	MNODES	11 I4	Number of nodes
	MSEGS		Number of stream segments
	NS		Number of streams
	LS		Number of information items in a stream = 18
	NESTK		(length of stream vector) NESTK = Ø if a starting point is not provided
			= number of constraints specified for a starting point
	NSTOP		= Ø if the program is to be executed until completion
			= 1 if the program is to be stopped after one iteration - to permit a check that the input has been read correctly

Card	Variable	Card Format	Description
	NSEGS		Number of segments (note: not streams) on which an upper and a lower bound on enthalpy is to be specified
	INF		Total number of entries in the vector KNF (see the description of KNF). = Ø if the option is not used
	INH		Total number of entries in the vector KNH (see the description of KNH). = Ø if the option is not used
	INA		Total number of entries in the vector KNA (see the description of KNA). = Ø if the option is not used
	JWRIT		= Ø if a complete output is to be printed = 1 if the output during optimization is to be suppressed.
Optional Cards (when NESTK ≠ 0)	KESTK (I) I=1, NESTK	9I8	List of codes for the initial constraints to be held (see the section on how to "code" a constraint)
Next set of Cards = 3xNS where NS is the number of streams	STPAR (I,J) I=1, 18 J=1, NS	8F8.1	Stream Information A set of 3 cards is necessary for every stream, with up to 8 entries per card
		For Any Stream J	
1st Card	STPAR (1,J)		= -2. for undefined flow of a process stream = -1. for undefined flow of a utility (steam or cooling water)

Card	Variable	Card Format	Description
			The undefined flow is identified in two different ways because the costing of the streams is different in each case.
			The cost of a process stream need be found once regardless of the number of units it passes through. A utility stream cost has to be determined at every heater or cooler that exists in which it is a utility.
	STPAR (2,J)		Inlet temperature = -1. if undefined
	STPAR (3,J)		Output temperature = -1. if undefined
	STPAR (4,J)		= Ø. Inlet pressure (Not used at present).
	STPAR (5,J)		= Ø. Outlet pressure (Not used at present).
	STPAR (6,J)		Inlet vapor fraction = -1. if the inlet temperature is undefined
	STPAR (7,J)		Outlet vapor fraction = -1. if the outlet temperature is undefined
	STPAR (8,J)		Liquid heat capacity
2nd Card	STPAR (9,J)		Vapor heat capacity
	STPAR (10,J)		Heat of vaporization
	STPAR (11,J)		Dew point temperature (Must be at least 1° greater than the bubble point temperature)
	STPAR (12,J)		Bubble point temperature
	STPAR (13,J)		Liquid-side film heat transfer coefficient
	STPAR (14,J)		Vapor-side film heat transfer coefficient

Card	Variable	Card Format	Description
	STPAR (15,J)		Two phase film heat transfer coefficient
	STPAR (16,J)		Cost coefficient ($/lb.* no. of operating hrs. per yr.)
3rd Card	STPAR (17,J)		Lower bound on flow. The lower bound must be set to quantity greater than zero (say 1% of total flow) if the stream is split. This helps avoid numerical difficulties.
	STPAR (18,J)		Upper bound on flows (Can be very large)
Next Card	A1	4F16.8	Step-size for Kuhn-Tucker test. Recommended value = 0.01 (change in scaled values for variables).
	A2		Initial step-size in scaled variable values for the complex method. Recommended value = 0.1
	A3		Minimum allowable approach temperature in degrees
	A4		Stopping criterion for the optimization (change in objective function). Recommended value = 0.00001
Next Card	CA	3F16.8	Cost multiplier for an exchanger (heater or cooler) x annual rate of return.
	EXPO		Exponent for the calculation of the cost of an exchanger, heater or cooler. Recommended value = 0.6.
	CB		Modification in the cost function of an exchanger, heater or cooler. Recommended value = 10. (Aids in avoiding the creation of local minima.)
			Cost of an exchanger, heater or cooler = CA*((Area + CB)**0.6 - CB**0.6)

Card	Variable	Card Format	Description
Next Set of Cards	ARK(I) I = 1,MNODES	6F10.2	This vector contains the information whether area is specified for a given node. = Ø. if area is not specified = value of area, if specified 6 area entries per card. For mixers and splitters, a value of zero is specified.
Optional Cards (when INA ≠ 0)	KNA(I) I = 1,INA	11I4	Gives information about exchangers, heaters and coolers that are to be related by areas Structure of KNA vector Entry 1 Item 1: Number of nodes in entry (say, n_1) Items 2,3,4,...n_1 + 1 Node numbers in ascending order. Repeat for entry 2 etc. New entries do not start a new card.
Optional Cards (when INF ≠ 0)	KNF(I) I = 1,INF		Gives information about unrelated segments which are to have equal flows. No segment flows are to be listed here as equal if they are already equal by the nature of the flowsheet. Structure of KNF vector Entry 1 Item 1: Number of stream segments in the entry (say, n_1) Item 2,3,...n_1 + 1 Stream segment numbers in ascending order. Repeat for entry 2 etc. New entries do not start a new card.
Optional Cards (when INH ≠ 0)	KNH(I) I = 1,INH	11I4	The description of this vector is the same as that of KNF with the exception that enthalpies are involved instead of flows.

Card	Variable	Card Format	Description
No. of Cards = MNODES (No. of nodes)	NARRAY(I,J) J = 1,11 I = 1, MNODES	11I4	One card for each node. Every card has 11 entries.

For node I which represents an exchanger heater or cooler

NARRAY(I,1)	Stream I.D. for the hot stream segments
NARRAY(I,2)	hot input segment number
NARRAY(I,3)	source node for hot input segment
NARRAY(I,4)	hot output segment number
NARRAY(I,5)	destination node for hot output segment
NARRAY(I,6)	cold input segment number
NARRAY(I,7)	source node for cold input segment
NARRAY(I,8)	cold output segment number
NARRAY(I,9)	destination node for cold output segment
NARRAY(I,10)	Type of node = 1 for a heat exchanger = 2 for a heater = 3 for a cooler
NARRAY(I,11)	Stream I.D. for the cold stream segments

For node I which represents a splitter

NARRAY(I,1)	Stream I.D. for the hot stream segment = 0 for a cold stream
NARRAY(I,2)	hot input segment number = -1 for a cold stream

Card	Variable	Card Format	Description
	NARRAY(I,3)		source node for hot input segment = -1 for a cold stream
	NARRAY(I,4)		the first output segment number
	NARRAY(I,5)		destination node for the first output segment
	NARRAY(I,6)		cold input segment number = -1 for a hot stream
	NARRAY(I,7)		source node for cold input segment = -1 for a hot stream
	NARRAY(I,8)		the second output segment number
	NARRAY(I,9)		destination node for the second output segment
	NARRAY(I,10)		Type of node = 4 for a splitter
	NARRAY(I,11)		Stream I.D. for the cold stream segments = 0 for a hot stream
			Note: the order for the output segments can be arbitrary.
	For node I representing a mixer		
	NARRAY(I,1)		Stream I.D. for the hot stream segments = 0 for a cold stream
	NARRAY(I,2)		the first input segment number
	NARRAY(I,3)		source node for the first input segment
	NARRAY(I,4)		hot output segment number = -1 for a cold stream
	NARRAY(I,5)		destination node for the hot output segment = -1 for a cold stream
	NARRAY(I,6)		second input segment number
	NARRAY(I,7)		source node for the second input segment

Card	Variable	Card Format	Description
	NARRAY(I,8)		cold output segment number = -1 for a hot stream
	NARRAY(I,9)		destination node for the cold output segment = -1 for a hot stream
	NARRAY(I,10)		type of node = 5 for a mixer
	NARRAY(I,11)		Stream I.D. for the cold stream segments = 0 for a hot stream Note: the order for the input segments can be arbitrary.
Optional Cards (when NSEGS ≠ 0)	ICON(I) I = 1,NSEG	11I4	This vector contains the segment numbers for which both the lower and upper bounds are to be specified for enthalpies.
Optional Cards (when NSEGS ≠ 0)	SECON(I,J) J = 1,2 I = 1,NSEGS	8F8.1	SECON(I,1) contains the lower bounds on enthalpies corresponding to the segment specified at the same I location in ICON. SECON(I,2) contains upper bounds. 8 entries per card with all the lower bounds specified first. Start with a new card for upper bounds.

Explanation About the Output From EROS

The output provides the following information:

1) A printout of the input.

2) The progress of optimization (optional).

3) Incidence matrix. The rows represent functions and columns represent variables.

4) Information about functions. The first column points to the row of the incidence matrix representing the function. The second column lists the node associated with the function. The third column indicates the type of equation involved.

TYPE = 1	Heat balance for an exchanger, heater or cooler
TYPE = 2,3	Material balances for a splitter
TYPE = 4	Material balance for a mixer
TYPE = 5	Heat balance for a mixer
TYPE > 10, TYPE < 0	Type is an inequality constraint code. (See Table 1.)

5) Identification of unknown variables. The first column stands for the segment involved. If the flow, temperature, or the split fraction associated with this segment is an unknown variable, the entry indicates the column number in the incidence matrix. Several variables may have the same column number if the system discovers they must be equal by heat or material balance. If the variable is known, a value of 0 is used.

6) Segment Information - Final values at optimal point.

7) Precedence order list - Together with 8, indicates how the equations were solved during the last optimization step.

8) Decisions and tears. If a variable is a decision, the corresponding value for its Function and MLIST columns are -1 and 1 respectively. If a tear variable is involved, the function it is assigned to is indicated in the corresponding column.

   MLIST = 2 represents explicit assignment.
   MLIST = 3 represents explicit assignment.

9) Objective function - Final value.

10) Exchanger Areas - Final values.

11) Information about time and number of iterations required.

COMPUTER PRINTOUT FOR A TYPICAL EXAMPLE
(FOR THE PROBLEM IN FIGURE 2).

*** INPUT ***

FL	T	A	SC
10000.0	100.0	1.0	100.0

MNODES	MSEGS	NS	LS	NESTK
6	16	5	18	0

NSTOP	NSEGS	INF	INH	INA
0	0	0	0	0

JWRIT
1

STPAR

10000.0	800.0	400.0	0.0	0.0	0.0	0.0	1.0
1.0	500.0	900.0	900.0	33.3	33.3	33.3	0.0
100.0	10000.0						
-1.0	600.0	600.0	0.0	0.0	1.0	0.0	1.0
1.0	800.0	600.0	600.0	100.0	100.0	100.0	0.4
0.0	90000.0						
-1.0	600.0	600.0	0.0	0.0	1.0	0.0	1.0

```
     1.0    800.0    600.0    600.0 10000.0 10000.0 10000.0      0.2
     0.0 90000.0
10000.0    200.0    500.0      0.0     0.0     0.0     0.0      1.0
     1.0    500.0    900.0    900.0   100.0   100.0   100.0      0.0
10000.0 10000.0
10000.0    200.0    500.0      0.0     0.0     0.0     0.0      1.0
     1.0    500.0    900.0    900.0    50.0    50.0    50.0      0.0
10000.0 10000.0
                A1                      A2                      A3                      A4
      0.01000000              0.10000000             18.00000000              0.00001000
                CA                    EXPO                      CB
     35.00000000              0.60000000              0.00000000

ARK
      0.00       0.00       0.00       0.00       0.00       0.00

NARRAY
  1   1   0   2   2  -1  -1   3   3   4   0
  1   2   1   4   4  11   0  12   5   1   4
  1   3   1   5   4  14   0  15   6   1   5
  1   4   2   6   0   5   3  -1  -1   5   0
  2   7   0   8   0  12   2  13   0   2   4
  3   9   0  10   0  15   3  16   0   2   5
```

*** OUTPUT ***

```
*****************************
*** OPTIMIZATION COMPLETE ***
*****************************
```

OBJECTIVE FUNCTION
3536.12

*** ACTIVE CONSTRAINTS AT SOLUTION ***
55 22

*** INCIDENCE MATRIX ***

1	0	0	0	0	0	0	0	1	0	0
0	1	0	0	0	0	0	0	1	0	0
1	0	0	0	0	1	0	1	0	0	0
0	1	0	0	0	0	1	0	1	0	0
1	1	1	0	0	0	0	0	0	0	0
1	1	1	0	0	1	1	0	0	0	0
0	0	0	1	0	0	0	1	0	0	0
0	0	0	0	1	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0

*** INFORMATION ABOUT FUNCTIONS ***

FUNC	NODE	TYPE
1	1	2
2	1	3
3	2	1
4	3	1
5	4	4
6	4	5
7	5	1
8	6	1
9	5	55
10	2	22

*** IDENTIFICATION OF UNKNOWN VARIABLES ***
SEG NO. FLOW TEMP SPLIT

1	0	0	0
2	1	0	10
3	2	0	10
4	1	6	0
5	2	7	0
6	3	0	0
7	4	0	0
8	4	0	0
9	5	0	0
10	5	0	0
11	0	0	0
12	0	8	0
13	0	0	0
14	0	0	0
15	0	9	0
16	0	0	0

*** SEGMENT INFORMATION ***

SEG NO.	FLOW	TEMP	PRESS	VAP. FRAC	ENTHAL	STREAM
1	10000.00	800.00	0.00	0.00	800.00	1.00
2	7349.95	800.00	0.00	0.00	800.00	1.00
3	2650.05	800.00	0.00	0.00	800.00	1.00
4	7349.95	391.83	0.00	0.00	391.83	1.00
5	2650.05	422.65	0.00	0.00	422.65	1.00
6	10000.00	400.00	0.00	0.00	400.00	1.00
7	0.00	600.00	0.00	1.00	1400.00	2.00
8	0.00	600.00	0.00	0.00	600.00	2.00
9	2500.00	600.00	0.00	1.00	1400.00	3.00
10	2500.00	600.00	0.00	0.00	600.00	3.00
11	10000.00	200.00	0.00	0.00	200.00	4.00
12	10000.00	500.00	0.00	0.00	500.00	4.00
13	10000.00	500.00	0.00	0.00	500.00	4.00
14	10000.00	200.00	0.00	0.00	200.00	5.00

15	10000.00	300.00	0.00	0.00	300.00	5.00
16	10000.00	500.00	0.00	0.00	500.00	5.00

*** PRECEDENCE ORDER LIST ***

VAR	FUNC
4	9
8	7
6	10
1	3
10	1
2	2
3	5
7	6
9	4
5	8
12	0
11	0
0	0

*** DECISIONS AND TEARS ***

VAR	FUNC	MLIST
12	-1	1

*** EXCH AREA ***

NO	AREA
1	0.000
2	496.110
3	145.855
4	0.000
5	0.000
6	220.821

*** INFORMATION ABOUT TIME AND ITERATIONS ***

	TIME (IN MILLISECS)
SETTING UP SOLUTION PROCEDURES	1477
ANALYSIS OF SOLUTION PROCEDURES	3205
MISCELLANEOUS TIME	2225
TOTAL TIME	6907

OBTAINING FIRST FEASIBLE POINT	2521

	NO OF ITERATIONS
SETTING UP SOLUTION PROCEDURES	4
ANALYSIS	39

APPENDIX B
DESCRIPTION OF THE PROGRAM

An abbreviated block diagram of the program is illustrated in Figure 23. Only the subroutines performing important functions are shown. A typical run of EROS takes place in the following manner. The main program makes a call to the subroutine EXEC. EXEC reads in the input data. Next, the routine START is called to perform an initialization wherein all the undefined parameters are set to a value of zero. Once this has been accomplished, EXEC performs the task of identifying all the unknown variables and numbering them sequentially by using the routines FLOCTS, HCNTS, KNOWN and CLEAR. EXEC issues directives to find the first solution procedure for the system. With the aid of FMATRX, an incidence matrix is created with rows representing model equations and columns representing the unknown variables. PRECED uses this matrix to find a solution procedure. At this point the control is transferred from EXEC to USPONT, a subroutine coordinating the two functions, optimization and creation of all subsequent solution procedures.

Every time a new solution procedure is created, USPONT transfers control to the optimization routine USOPT. USOPT makes use of the complex method represented by OPT, and the initialization for the optimization is performed by ICOPT. OPT hands back different sets of

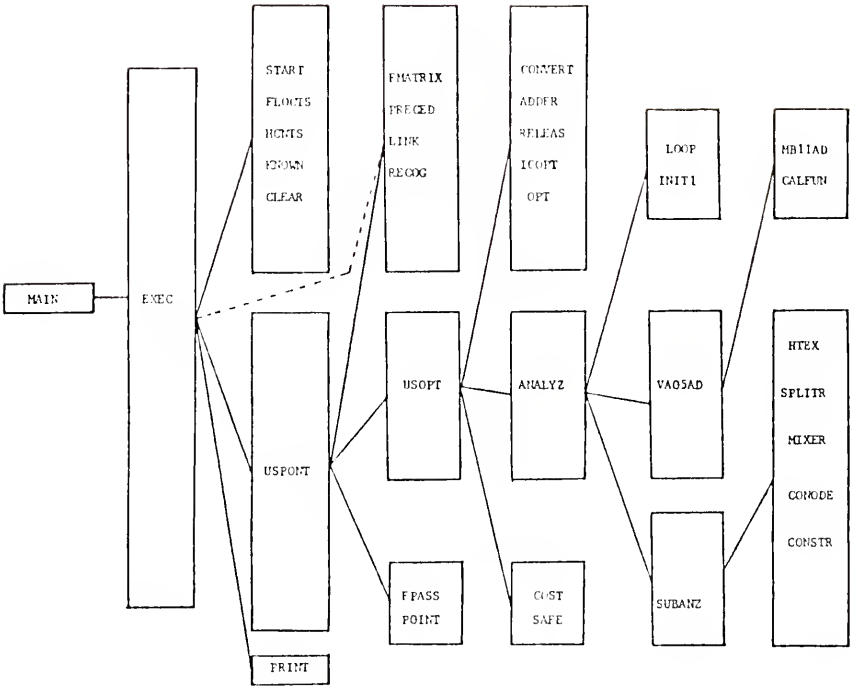


Fig. 23. Description of the Program EROS

decision variable values to USOPT. For each set of decision variable values, model equations are solved and the objective function found. USOPT invokes the use of ANALYZ in order to solve the model equations. The convergence of the iteration variables in ANALYZ is aided by the Harwell Library subroutine VA05AD. Constraint violations are checked for in SUBANZ. If no constraints are violated, USOPT requests for the total system cost via the routine COST. If constraint violations occur, control is transferred back to USPONT. A decision regarding which inequality constraints are to be present as equalities in the equation set is made in POINT, and then a new incidence matrix and consequently a new solution procedure is found in USPONT. Optimization is started once again, using USOPT.

When the stopping criteria for optimization are satisfied, control is transferred to EXEC from USOPT through USPONT. An output is requested by a call to PRINT and the run terminated.

A brief description of all the subroutines and function subprograms used in EROS is presented next. The modules are presented in an alphabetical order and unless they are specified explicitly as function subprograms, they represent subroutines.

ACYCLIC

This routine helps in finding a solution procedure. It consists of the acyclic algorithm.

ADDER

When a new solution procedure is found, constraints in the equation set corresponding to slack variables whose values are zero, are held tight.

AHELP

Using the area of a heat-exchanger zone, and the user specified constants, cost is calculated in this subroutine.

ANALYZ

Groups of equations that can be solved without iterating and calculations involving an iteration loop are identified so that the model equations may be solved.

Subroutines used: LOOP
INIT1
VA05AD
SUBANZ
FILLX

AREA

Areas are calculated for different phase zones and subsequently the cost for an exchanger is evaluated.

Subroutine used: AHELP

BUBPT

This is a function subprogram that discovers the bubble-point temperature for a stream segment.

CALFUN

This routine is called by VA05AD to provide the value of the error function.

Subroutine used: SUBANZ

CENTRE

A centroid is found for the points generated by the complex method.

CLEAR

The unknown variables are numbered sequentially.

Subroutine used: CLEAR1

CLEAR1

An aid to the routine CLEAR.

COLOC

This subroutine is used to find the value of a slack variable from the current best point.

CONODE

The unknown variable involved in a constraint present as an equality is calculated by this routine.

Subroutines used: FILLUP
ENTHAL
ENSLAK
TEMP
COLOC

Function Subprograms Used: DEWPT

CONSTR

A check is made to see if any constraints are violated. If none are violated the slack values for all the constraints are stored.

Subroutines used: ENTHAL
TEMP
ENSLAK

Function Subprograms used: DEWPT
BUBPT

CONVRT

In this routine, an array that provides information about slack variables associated with constraints in the equation set, is created.

COST

Once all the model equations are solved, the costs associated with exchangers and the utility streams are added together in this subroutine to provide the value of the objective function.

Subroutine used: AREA

DEWPT

This is a function subprogram that discovers the dew-point temperature for a stream segment.

ENBALH

During the setting up of an incidence matrix, this routine helps in creating a row for the heat-balance of an exchanger.

ENSLAK

This routine helps in treating the temperature constraints in terms of enthalpies

Subroutines used: ENTHAL
TEMP

ENTHAL

Used for calculating the enthalpy and vapor fraction of a segment given its temperature. For a pure component enthalpy is calculated given both temperature and vapor fraction.

Function Subprograms used: BUBPT
DEWPT

EQS

Helps relate flows, enthalpies or areas for reliability analysis.

EXEC

This is an executive routine to which all the input data are provided. Calls are made to several subroutines for the execution of

the program. The directive for providing the final output is also issued by EXEC.

Subroutines used: START
FLOCTS
HCNTS
KNOWN
CLEAR
FMATRX
SETUP
PRECED
LINK
ANALYZ
COST
USPONT
PRINT

FILLUP

Once an unknown variable is calculated, its value is stored for all the segments it occurs in.

FILLX

The value of the decision variable is stored for all the segments this variable occurs in.

Subroutine used: TEMP

FLOCTS

All the segments having a common flow-rate are identified and characterized by a unique number in this routine.

Subroutine used: EQS

FMATRX

An incidence matrix is created in which the rows represent the heat and material balances and inequality constraints converted to equalities. The columns represent the unknown variables.

Subroutine used: SETUP

FPASS

This routine is used when a new solution procedure is to be created by replacing one of the constraints present in the equation set by the most recently violated constraint.

HCNTS

All the segments having a common enthalpy are characterized by a unique number in this routine.

Subroutine used: EQS

HTEX

Any unknown variable in the heat-balance equation of the heat-exchanger is calculated by this subroutine. The error function associated with the heat-balance equation is also calculated.

Subroutine used: FILLUP
TEMP

HTTC

This function subprogram is used to calculate the overall heat-transfer coefficient based on the film heat-transfer coefficients of the streams involved.

ICOPT

Values for decision variables and other parameters involved in optimization are initialized.

Subroutines used: COLOC
SORT

INIT1

Initializes parameters for the use of VA05AD.

KNOWN

In this routine the known values for flows and enthalpies are taken into account to help discover what the unknown variables are.

Subroutine used: ENTHAL

LINK

An array is created with entries that keep track of the segments that have the same flow or enthalpy.

LOOP

Identifies all the iteration loops for ANALYZ.

MB11AD

A Harwell Library subroutine called by VA05AD.

MIXER

The unknown variables involved in the heat and material balances of a mixer and the error functions are calculated by this subroutine.

Subroutines used: TEMP
FILLUP

OPT

This routine consists of the modified complex algorithm for optimization.

Subroutine used: CENTRE

POINT

In this routine a decision is made regarding what constraints are to be incorporated into the equation set to obtain a new solution procedure.

PRECED

A solution procedure is discovered for the given set of equations represented in the matrix form. The decisions and tear variables are established.

Subroutine used: ACYCLIC

PRINT

Provides an output.

RECOG

An array is created in which information concerning the unknown variables (whether they represent flows, enthalpies, or split fractions) is stored.

RELEAS

Slack variables corresponding to constraints in the equation set are identified if they are to be treated as search coordinates.

SAFE

The value of the objective function corresponding to the best current point is stored.

SETUP

Creates an incidence matrix with rows represented only by the heat and material balance equations.

Subroutine used: ENBALH

SORT

A routine that creates an array containing information about slack variables that are to be used as search coordinates.

SPLITR

Any unknown variables in the material balance equations of a splitter, and the error functions involved, are calculated by this subroutine.

Subroutine used: FILLUP

START

Initializes all the unspecified parameters at the start of execution.

Subroutine used: EQS

SUBANZ

For the calculation of an unknown variable, a call is made to the unit model routines associated with the variable. After all the calculations have been completed, CONSTR is called to check for constraint violations.

Subroutines used:
HTEX
SPLITR
MIXER
CONODE
CONSTR

TEMP

The temperature and vapor fraction of a segment are calculated given its enthalpy.

USOPT

Optimization is performed using the strategy followed in this study.

Subroutines called:
CONVRT
ADDER
RELEAS
ICOPT
OPT
ANALYZ
COST

USPONT

The algorithm for finding a new solution procedure is coordinated with the optimization strategy.

Subroutines used:
SETUP
FMATRX
PRECED
LINK
RECOG
USOPT
FPASS
POINT
PRINT

VA05AD

A Harwell Library subroutine that performs a least-square fit on a set of non-linear equations.

Subroutines used:
MB11AD
CALFUN

In addition, a system subroutine (CPUTIM) on IBM 360/67 ar Carnegie-Mellon University was used. This routine provided the elapsed CPU time in milliseconds. It was called by the following subroutines:

EXEC
ANALYZ
USPONT

LITERATURE CITED

Avriel, M., and A.C. Williams, "An Extension of Geometric Programming with Applications in Engineering Optimization," J. of Engr. Math., 5, 187 (1971).

Boas, A.H., "Optimization via Linear and Dynamic Programming," Chem. Eng., 70, 85 (1963).

Bragin, M.S., "Optimization of Multistage Heat Exchanger Systems," Ph.D. dissertation, New York University (1966).

Brosilow, C. and L. Lasdon, "A Two Level Optimization Technique for Recycle Processes," AIChE-IChemE Symp. Ser. No. 4, 75 (1965).

deBrosse, C.J., and A.W. Westerberg, "A Feasible-Point Algorithm for Structured Design Systems in Chemical Engineering," AIChE J., 19, 251 (1973).

Director, S.W., Hachtel, G.D. and L.M. Vidigal, "Computer Efficient Yield Estimation Procedure Based on Simplicial Approximation," IEEE Trans. Circuits and Systems, to be published (1978).

Fan, L.T., and C.S. Wang, "The Discrete Maximum Principle," Wiley, N.Y. (1964).

Geoffrion, A.M., "Elements of Large-Scale Mathematical Programming, 1" the Rand Corporation, R-481-PR, Santa Monica, Calif., November (1969).

Grossman, I.E., and R.W.H. Sargent, a personal communication (1977).

Guthrie, K.M., "Capital Cost Estimating," Chem. Eng., pp 114-142, March 24 (1969).

Hadley, G., "Linear Programming," Addison-Wesley, Reading, Massachusetts, pp 158-162 (1963).

Henley, E.J., and R.A. Williams, "Graph Theory in Modern Engineering," Academic Press, N.Y. (1973).

Himmelblau, D.M., "Optimal Design via Structural Parameters and Non-linear Programming," U.S.-Japan Joint Seminar on Application of Process Systems Engineering to Chemical Technology Assessment, Kyoto, Japan, June 23-27, 1975.

Hutchison, H.P., and C.F. Shewchuk, "Computational Method for Multiple Distillation Towers," Trans. Instu. Chem. Eng., 52 (4), p 325 (1974).

Hwa, C.S., "Mathematical Formulation and Optimization of Heat Exchanger Networks Using Separable Programming," Symp. Series No. 4, pp 101-106, AIChE-IChemE Joint Meeting, London (1965).

Ichikawa, A., and L.T. Fan, "Optimal Synthesis of Process Systems. Necessary Condition for Optimal System and its Use in Synthesis of Systems," Chem. Eng. Sci., 28, 357 (1973).

Kubicek, M., V. Hlavacek and F. Prochaska, "Global Modular Newton-Raphson Technique for Simulation of an Interconnected Plant Applied to Complex Rectification Columns," Chem. Engr. Sci., 31, 277 (1976).

Kuhn, H.W., and A.W. Tucker, "Non-Linear Programming," in Proc. 2d Berkeley Symp. on Math. Statistics Prob., 481, Univ. of Calif. Press, Berkeley (1951).

Lasdon, L.A., "Optimization Theory for Large Systems," Macmillan Co., New York (1970).

Lee, W., J.H. Christensen and D.F. Rudd, "Design Variable Selection to Simplify Process Calculations," AIChE J., 12, 1104 (1966).

Lee, K.F., A.H. Masso and D.F. Rudd, "Branch and Bound Synthesis of Integrated Process Designs," Ind. Eng. Chem. Fundamentals, 9, 48 (1970).

Leigh, M.J., G.D. Jackson and R.W.H. Sargent, "SPEED-UP -- A Computer-Based System for the Design of Chemical Processes," paper presented at CAD-74, Imperial College, London, England, Sept. 24-27, 1974.

Masso, A.M., and D.F. Rudd, "The Synthesis of System Designs - II Heuristic Structuring," AIChE J., 15, 10 (1969).

McGalliard, R.L., and A.W. Westerberg, "Structural Sensitivity Analysis in Design Synthesis," Chem. Eng. J., 4 (4), 127 (1972).

Mishra, P.N., L.T. Fan and L.E. Erickson, "Biological Wastewater Treatment System Design," Can. J. Chem. Eng., 51, 694 (1973).

Nishida, N., Y.A. Liu and L. Lapidus, "Studies in Chemical Process Design and Synthesis: III. A Simple and Practical Approach to the Optimal Synthesis of Heat Exchanger Networks," AIChE J., 23, 77 (1977).

Osakada, K., and L.T. Fan, "Synthesis of an Optimal Large-Scale Interconnected System by Structural Parameter Method Coupled with Multilevel Technique," Can. J. Chem. Eng., 51, 94 (1973).

Perry, R.H., and C.H. Chilton, Chemical Engineering Handbook, 5th Edition, McGraw-Hill, New York, pp 10-39 to 10-42 (1973).

Takamatsu, T., I. Hashimoto, H. Nishitani and S. Tomita, "The Optimal Design of Large Complex Chemical Processes - Development of Max-Sensitive Method," Chem. Engng. Sci., 31, 705 (1976).

Takamatsu, T., I. Hashimoto and H. Ohno, "Optimal Design of a Langle Complex System from the View Point of Sensitivity Analysis," Ind. Engng. Chem. Proc. Des. and Dev., 9, 368 (1970).

Umeda, T., A. Hirai and A. Ichikawa, "Synthesis of Optimal Processing System by an Integrated Approach," Chem. Eng. Sci., 27, 795 (1972).

Umeda, T., and A. Ichikawa, "A Modified Complex Method for Optimization," Ind. Engng. Chem. Proc. Des. and Dev., 10, 229 (1971).

Westbrook, G.T., "Use This Method to Size Each Stage for Best Operation," Hydrocarbon Processing, 40, 201 (1961).

Westerberg, A.W., and C.J. deBrosse, "An Optimization Algorithm for Structured Design Systems," AIChE J., 19, 335 (1973).

BIOGRAPHICAL SKETCH

Jigar Shah, the eldest son of Hemlata and Vinubhai Shah, was born on December 27, 1951, in Ahmedabad, India. He graduated from St. Peter's Boys' High School in Panchgani in December 1967. From July 1968 to May 1973 he studied chemical engineering at Indian Institute of Technology, Madras. After the receipt of the B.Tech. degree, he was accepted as a graduate student at the University of Florida. He completed his M.S. in chemical engineering in December 1974 and proceeded to work for a doctoral degree. While working on doctoral research he accompanied his advisor, Dr. A. Westerberg, to Computer-Aided Design Centre, Cambridge, U.K., from January 1975 to June 1975, and to Carnegie-Mellon University, Pittsburgh, from September 1976 to January 1978. Jigar Shah is currently a member of the American Institute of Chemical Engineers.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Arthur W. Westerberg, Chairman
Professor of Chemical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

John P. O'Connell
Professor of Chemical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Hugh D. Ratliff
Professor of Industrial and Systems Engineering

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate Council, and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

March 1978

Dean, College of Engineering

Dean, Graduate School